SISTEM KENDALI SUHU OTOMATIS UNTUK PROSES STERILISASI SUSU DENGAN METODE KONVENSIONAL MENGGUNAKAN KONTROL PID BERBASIS ARDUINO MEGA 2560 DENGAN TELEMETRI

ABDUL ROZAK
5215110243

Skripsi Ini Ditulis Untuk Memenuhi Sebagian Persyaratan
Dalam Memperoleh Gelar Sarjana

PROGRAM STUDI PENDIDIKAN TEKNIK ELEKTRONIKA
JURUSAN TEKNIK ELEKTRO
FAKULTAS TEKNIK
UNIVERSITAS NEGERI JAKARTA
2015

ABSTRAK

Abdul Rozak, *Sistem Kendali Suhu Otomatis untuk Proses Sterilisasi Susu dengan Metode Konvensional Menggunakan Kontrol PID Berbasis Arduino Mega 2560 dengan Telemetri*. Skripsi. Jakarta, Program Studi Pendidikan Teknik Elektronika, Jurusan Teknik Elektro, Fakultas Teknik, Universitas Negeri Jakarta, 2015. Dosen Pembimbing, Drs. JUSUF BINTORO, MT. dan MUHAMMAD YUSRO, S.Pd., MT.

Tujuan dari penelitian ini adalah merancang dan membuat sistem kontrol suhu otomatis untuk menjaga kestabilan suhu pada saat proses sterilisasi susu dengan metode konvensional dilakukan dan proses tersebut dapat dipantau dan dikontrol dari jarak jauh (telemetri) dengan *smartphone* Android melalui koneksi *Bluetooth.*

Penelitian ini dilakukan menggunakan Metode Penelitian dan Pengembangan (*Research and Development*) yang meliputi perencanaan, analisis kebutuhan, perancangan, pengujian, dan implementasi sistem. Kebutuhan sistem yang diperlukan pada penelitian ini adalah: sensor suhu yang dapat mengukur suhu pada objek secara tak sentuh, sistem kontrol suhu yang stabil, tepat waktu, dan dapat dikontrol dan dipantau dari jarak jauh (telemetri).

Hasil penelitian ini menunjukkan *Sistem Kontrol Suhu Otomatis untuk Proses Sterilisasi Susu dengan Metode Konvensional Berbasis Arduino Mega 2560 dengan Telemetri* yang telah dirancang, direalisasikan, dan diuji dapat menjaga kestabilan suhu pada saat proses sterilisasi susu dilakukan yaitu pada suhu 120^0C selama 15 menit dengan persentasi *error* sebesar 0,26% dengan parameter KP = 7, KI = 7, KD = 40. Proses sterilisasi susu tersebut juga dapat dipantau dan dikontrol dari jarak jauh secara telemetri dengan *smartphone* Android melalui koneksi *Bluetooth* dengan radius maksimal 10 meter.

Kata kunci: Sistem Kendali Suhu Otomatis, Sterilisasi Susu dengan Metode Konvensional, Kontrol PID, Telemetri, *Bluetooth*, Arduino Mega 2650, Sensor Suhu Inframerah

ABSTRACT

Abdul Rozak, Automatic Temperature Control System for Milk Sterilization Process with Conventional Methods Using PID Control Based Arduino Mega 2560 with Telemetry. Thesis. Jakarta, Electronics Engineering Education Studies Progra, Department of Electrical Engineering, Faculty of Engineering, State University of Jakarta, 2015. Supervisor, Drs. JUSUF BINTORO, MT. and MUHAMMAD YUSRO, S.Pd., MT.

The aim of this study was to design and create automatic temperature control system to maintain a stable temperature during the process of sterilization of milk by conventional methods performed and the process can be monitored and controlled remotely by telemetry with Android smartphones via Bluetooth connection.

This research was conducted using the method of Research and Development, which includes planning, requirements analysis, design, testing, and implementation of the system. System requirements necessary in this study are: a temperature sensor that can measure the temperature on the object do not touch, temperature control system is stable, timely, and can be controlled and monitored from a distance (telemetry).

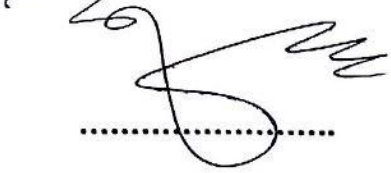
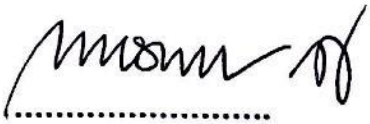
The results showed System Automatic Temperature Control for Process Sterilization Milk with Conventional Methods Based Arduino Mega 2560 with telemetry that has been designed, realized and tested can maintain a stable temperature during the process of sterilization of milk made at a temperature of 120^0C for 15 minutes with a percentage error of 0.26% with the parameters KP = 7, KI = 7, KD = 40. The milk sterilization process can also be monitored and controlled remotely by telemetry with Android smartphones via Bluetooth connection with a maximum radius of 10 meters.

Keywords: Automatic Temperature Control Systems, Sterilization Milk with conventional methods, PID control, Telemetry, Bluetooth, Arduino Mega 2650, Infrared Temperature Sensor

HALAMAN PENGESAHAN

NAMA DOSEN	TANDA TANGAN	TANGGAL
Drs. Jusuf Bintoro, MT **(Dosen Pembimbing I)**		23/12/2015
Muhammad Yusro, S.Pd., MT **(Dosen Pembimbing II)**		29/12/2015

PENGESAHAN PANITIA UJIAN SKRIPSI

NAMA DOSEN	TANDA TANGAN	TANGGAL
Drs. Pitoyo Yuliatmojo, MT **(Ketua Penguji)**		22/12/2015
Efri Sandi, S.Pd., MT **(Anggota Penguji)**		23/12/2015
Drs. Wisnu Djatmiko, MT **(Anggota Penguji)**		11/01/2016

HALAMAN PERNYATAAN

Dengan ini saya menyatakan bahwa:

1. Karya tulis skripsi/komprehensif/karya inovatif saya ini adalah asli dan belum pernah diajukan untuk mendapatkan gelar akademik, baik di Universitas Negeri Jakarta maupun di perguruan tinggi lain.
2. Karya tulis adalah murni gagasan dan penelitian saya sendiri dengan arahan dosen pembimbing.
3. Dalam karya tulis ini terdapat karya atau pendapat yang telah ditulis atau dipublikasi orang lain, kecuali secara tertulis dengan jelas atau dicantumkan sebagai acuan dalam naskah dengan disebutkan nama pengarang dan dicantumkan dalam daftar pustaka.
4. Pernyataan ini saya buat dengan sesungguhnya dan apabila dikemudian hari terdapat penyimpangan dan ketidakbenaran dalam pernyataan ini, maka saya bersedia menerima sanksi akademik berupa pencabutan gelar yang telah diperoleh karena karya tulis ini, serta sanksi lainnya sesuai dengan norma yang berlaku di Universitas Negeri Jakarta.

Jakarta, 25 November 2015

Yang membuat Pernyataan

Abdul Rozak

NIM: 5215110243

KATA PENGANTAR

Puji syukur penulis sampaikan kepada Allah Subhanahu Wa Ta'ala atas segala Karunia dan Rahmat-Nya sehingga skripsi ini dapat terselesaikan. Penulisan skripsi ini dilakukan dalam rangka memenuhi salah satu syarat untuk mencapai gelar Sarjana Pendidikan Teknik Elektronika Jurusan Teknik Elektro Fakultas Teknik Universitas Negeri Jakarta. Saya menyadari bahwa tanpa bantuan dan bimibngan dari berbagai pihak akan sangat sulit bagi saya untuk menyusun skripsi ini. Oleh Karena itu, saya mengucapkan terima kasih kepada:

1) Bapak Drs. Jusuf Bintoro, MT selaku dosen pembimbing I.
2) Bapak Muhammad Yusro, S.Pd., MT selaku dosen pembimbing II.
3) Bapak Prasetyo Wibowo, S.T., M.Eng selaku pembimbing akademik
4) Kedua orang tua dan keluarga saya yang telah memberikan kasih sayang yang tidak ternilai hargarnya dan juga atas doa yang tidak pernah terhenti diucapkan.
5) Rekan-rekan mahasiswa/i Pendidikan Teknik Elektronika angkata 2011 yang telah membantu dan memberi dukungan dalam penyusunan skripsi ini.

Akhir kata, semoga Allah Subhanhu Wa Ta'ala membalas segala kebaikan semua pihak yang telah membantu penyusunan skripsi ini dengan balasan yang lebih baik. Semoga skripsi ini membawa manfaat yang besar bagi pengembangan ilmu pengetahuan dan teknologi.

Jakarta, 25 November 2015

Abdul Rozak

5215110243

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR LAMPIRAN

BAB I

PENDAHULUAN

1.1 Latar Belakang Masalah

Susu merupakan bahan pangan yang tersusun oleh zat-zat makanan dengan proporsi seimbang, bernilai gizi tinggi, mudah dicerna dan mengandung semua unsur makanan yang dibutuhkan oleh manusia. Dengan kandungan nutrisinya yang lengkap, susu merupakan media yang baik bagi pertumbuhan mikroorganisme, oleh karena itu susu mudah mengalami kerusakan. Untuk mencegah kerusakan susu oleh mikroba maka dilakukan berbagai upaya pengawetan seperti sterilisasi.

Sterilisasi susu merupakan proses yang dilakukan terhadap susu cari segar yang diolah menggunakan pemanasan pada suhu tinggi dan stabil dalam waktu yang singkat untuk membunuh seluruh mikroorganisme yang terdapat di dalam susu sehingga memiliki kualitas yang baik. Sterilisasi susu akan memperpanjang daya simpan, tetapi jika tidak disterilkan pada temperatur dan waktu yang tepat akan terjadinya kerusakan nutrisi yang terkandung di dalam susu.

Dalam kasus ini penggunaan sistem kendali otomatis dapat dijadikan solusi yang tepat dalam menjaga kestabilan suhu yang diinginkan, sehingga akan menghasilkan susu sterilisasi dengan kualitas yang baik. Mengingat proses sterilisasi susu berhubungan dengan panas yang tinggi dan berbahaya, maka dibutuhkan sebuah cara agar pemantauan dan pengontrolan proses sterilisasi susu tersebut dapat dilakukan dari jarak jauh. Untuk itulah penulis akan mengangkat sebuah judul pada penelitan ini yaitu, *"Sistem Kendali Suhu Otomatis untuk Proses Sterilisasi Susu dengan Metode Konvensional Menggunakan Kontrol PID Berbasis Arduino Mega 2560 dengan Telemetri".*

1.2 Identifikasi Masalah

Dari latar belakang masalah di atas maka dapat diidentifikasi masalah sebagai berikut:

1. Susu merupakan bahan pangan yang mudah rusak akibat mikroorganisme
2. Perlunya proses sterilisasi susu agar mikroorganisme yang terdapat di dalam susu menjadi mati dan susu dapat disimpan dalam jangka waktu yang lama.
3. Perlunya sistem kendali suhu otomatis yang dapat menjaga kestabilan suhu pada saat proses sterilisasi susu dilakukan.
4. Perlunya penggunaan telemetri pada sistem kendali suhu otomatis agar sistem dapat dipantau dan dikontrol dari jarak jauh mengingat proses sterilisasi susu ini berhubungan dengan temperatur yang tinggi dan berbahaya

1.3 Pembatasan Masalah

Dari identifikasi masalah di atas maka dapat dibuat batasan-batasan masalah sebagai berikut:

1. Susu yang digunakan adalah air susu segar.
2. Proses sterilisasi yang digunakan adalah proses sterilisasi dengan metode konvensional.
3. Menggunakan sistem kendali suhu otomatis berbasis Arduino Mega 2560.
4. Penggunaan sensor suhu MLX90614 untuk mendeteksi suhu pada *plant*.
5. Penggunaan pemanas air 600Watt sebagai alat pemanas untuk steriliasi susu.
6. Penggunaan algoritma kontrol PID pada sistem kendali suhu.
7. Penggunaan *Bluetooth* HC-06 sebagai perangkat *telemetri*

1.4 Perumusan Masalah

Dari pembatasan masalah di atas maka dapat dibuat rumusan masalah sebagai berikut:

1. Bagaimana cara merancang sistem kendali suhu otomatis yang dapat menjaga kestabilan suhu sebesar 120^0C selama 15 menit untuk proses sterilisasi susu?
2. Bagaimana cara merancang sistem kendali suhu otomatis tersebut agar dapat dipantau dan dikontrol dari jarak jauh (telemetri) dengan menggunakan *smartphone* Android?.

1.5 Tujuan Penelitian

Tujuan dari penelitian ini adalah merancang dan membuat sistem kendali suhu otomatis yang dapat menjaga kestabilan suhu pada proses sterilisasi susu dengan metode konvensional dan proses tersebut dapat dipantau dan dikontrol dari jarak jauh (telemetri) dengan menggunakan *smartphone* Android melalui koneksi *Bluetooth.*

1.6 Kegunaan penelitian

1. Pemanfaatan ilmu sistem kendali otomatis di dalam dunia industri.
2. Dapat membuat sebuah prototipe sistem kendali suhu pada industri.
3. Untuk meningkatkan pengetahuan peneliti tentang ilmu sistem kendali otomatis dalam dunia industri.

BAB II

KERANGKA TEORETIK, KERANGKA BERPIKIR, DAN HIPOTESIS PENELITIAN

2.1 Kerangka Teoretik

2.1.1 Susu

Susu merupakan bahan makanan yang bernilai gizi tinggi yang diperoleh dari hasil pemerahan hewan mamalia seperti sapi, kerbau, kuda, kambing dan unta. Komponen penting dalam air susu adalah protein, lemak, vitamin, mineral, laktosa serta enzim-enzim dan beberapa jenis mikroba yang bermanfaat bagi kesehatan sebagai probiotik (Pusdatin, 2013).

2.1.2 Sterilisasi Susu dengan Metode Konvensional

Steriliasi adalah perlakuan untuk menjadikan suatu bahan atau benda bebas dari mikroorganisme dengan cara pemanasan, penyinaran, atau dengan zat kimia yang mematikan mikroorganisme hidup maupun sporanya (KBBI, Sterilisasi, 2015).

Sterilisasi susu dengan metode konvensional adalah proses pengawetan susu dengan cara dipanaskan pada suhu 120-125^{0}C selama 11-15 Menit (Susanti, 2015). Sterilisasi juga harus diikuti oleh pengemasan yang baik dan pengaturan suhu penyimpanan. Dalam kemasan yang belum dibuka, umur simpan susu ini bisa mencapai tujuh bulan dalam ruangan yang kering dan bersih.

2.1.3 Manfaat Sterilisasi Susu

Tujuan utama proses sterilisasi susu adalah untuk membunuh mikroorganisme penyebab penyakit atau penyebab kerusakan bahan pangan,

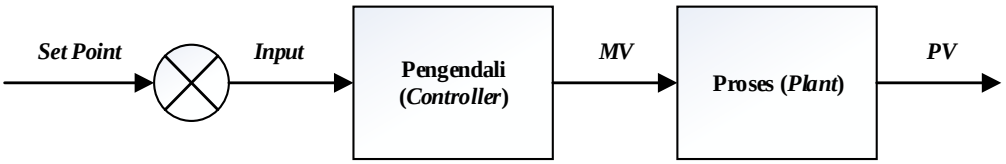
sehingga produk susu yang dihasilkan menjadi lebih awet dan aman (Chavan, Chavan, Khedkar, & Jana, 2011).

2.1.4 Sistem Kendali

Sistem kendali merupakan serangkaian komponen fisik yang berfungsi untuk mengendalikan kondisi lingkungan dari sebuah *plant* menjadi kondisi atau respon yang diinginkan, sebagai contoh: kendali suhu ruang, mesin cuci, robot, pesawat, dan lain sebagainya (Katsuhiko & Ogata, 1997). Secara umum sistem kendali terbagi menjadi dua jenis, yaitu sistem kendali lingkar terbuka (*open loop control system*) dan sistem kendali lingkar tertutup atau umpan-balik (*feedback control system*).

2.1.4.1 Sistem Kendali Lingkar Terbuka

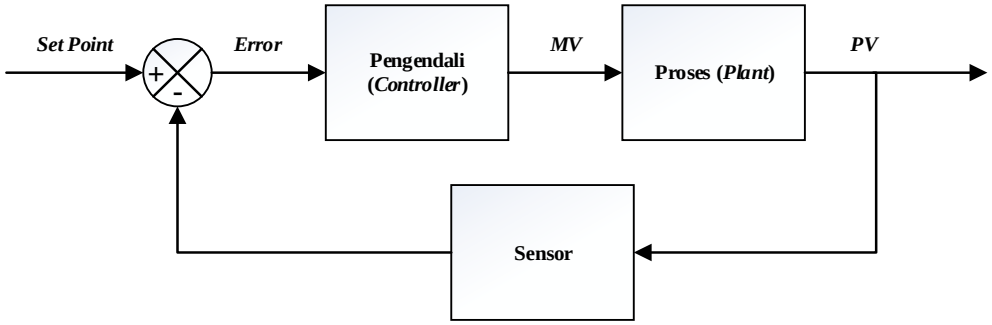
Sistem kendali lingkar terbuka (*open loop control system*) adalah sistem kendali di mana tidak terdapat elemen yang mengamati keluaran yang terjadi yang kemudian dibandingkan dengan nilai referensi yang dinginkan oleh sistem sehingga keluaran sistem tidak akan mempengeharuhi masukkan sistem. Diagram blok sistem kendali lingkar terbuka (*open loop*) dapat dilihat pada gambar 2.1:



Gambar 2.1 Sistem Kendali Lingkar Terbuka (*Open Loop*) *(Katsuhiko & Ogata, 1997)*

2.1.4.2 Sistem Kendali Lingkar Tertutup atau Umpan Balik

Sistem kendali lingkar tertutup atau umpan-balik (*Feedback Control System*) adalah sistem kendali yang mempunyai elemen umpan balik yang berfungsi untuk mengamati keluaran yang terjadi lalu dibandingkan dengan nilai referensi yang diinginkan sistem kendali. Diagram blok sistem kendali lingkar tertutup atau umpan balik (*feedback control system*) dapat dilihat pada gambar 2.2:



Gambar 2.2 Sistem Kendali Lingkar Tertutup atau Umpan Balik (*Feedback Control System*) *(Katsuhiko & Ogata, 1997)*

2.1.4.3 Definisi-definisi Sistem Kendali

Dalam sistem kendali ada beberapa definisi-defenisi yang sering digunakan dalam praktiknya antara lain:

1. *Set Point* (SP) adalah harga atau nilai yang ingin dicapai oleh sistem (contoh: nilai suhu yang dibutuhkan untuk proses sterilisasi susu).
2. *Manipulated Variable* (MV) adalah harga atau nilai yang diatur agar proses menjadi stabil. *Manipulated Variable* biasanya dihubungkan dengan input aktuator (contoh: rangkaian penggerak pemanas air)

3. *Process Variable* (PV) adalah sinyal hasil pemantauan terhadap proses atau *plant*. *Proses Variable* umumnya adalah hasil dari pembacaan dari suatu sensor (contoh: Sensor suhu).
4. *Error* adalah selisih antara *Set Point* dengan *Proses Variable.*
5. *Plant* adalah objek yang akan dikendalikan oleh sistem kendali (contoh: temperatur pada air).

2.1.4.4 Elemen-elemen Sistem Kendali

Secara umum elemen sistem kendali dibagi dalam 4 bagian yaitu, sensor dan transduser, *error detector,* penggerak, dan penguat (Katsuhiko & Ogata, 1997).

1. **Sensor dan Transduser**

Sensor digunakan sebagai elemen yang langsung mengadakan kontak dengan yang diukur sedangkan transduser berfungsi untuk mengubah besaran fisis yang diukur menjadi besaran fisis lainnya. Sensor pada umumnya mengubah besaran-besaran fisis menjadi besaran listrik seperti tekanan, temperatur, aliran, posisi dan lain-lain. Pada penelitian Sensor yang digunakan adalah termometer inframerah untuk mengukur suhu pada *plant.*

2. ***Error Detector***

Error detector berfungsi untuk mengukur *error* (kesalahan) yang terjadi antara keluaran aktual dan keluaran yang diinginkan. *Error detector* dalam penelitian ini menggunakan Mikrokontroler Arduino.

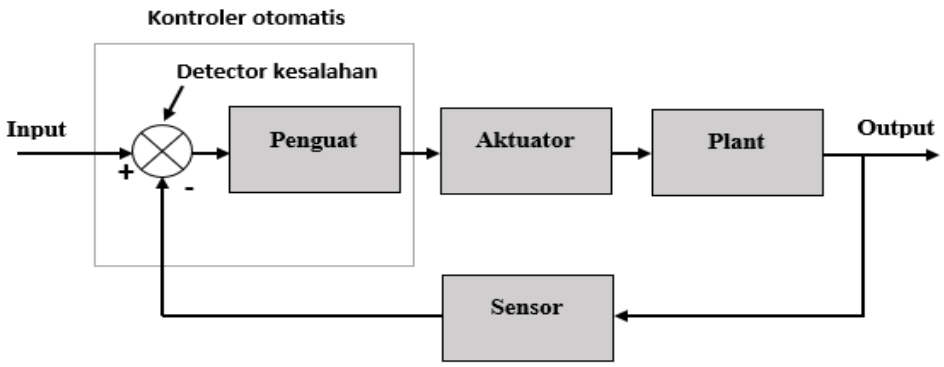
3. **Penggerak**

Alat ini berfungsi untuk mengendalikan aliran energi ke sistem yang dikendalikan. alat ini disebut juga elemen pengendali akhir, misalnya rangkaian

penggerak pemanas listrik. Elemen keluaran ini harus mempunyai kemampuan untuk menggerakkan beban ke suatu harga yang diinginkan.

4. Penguat

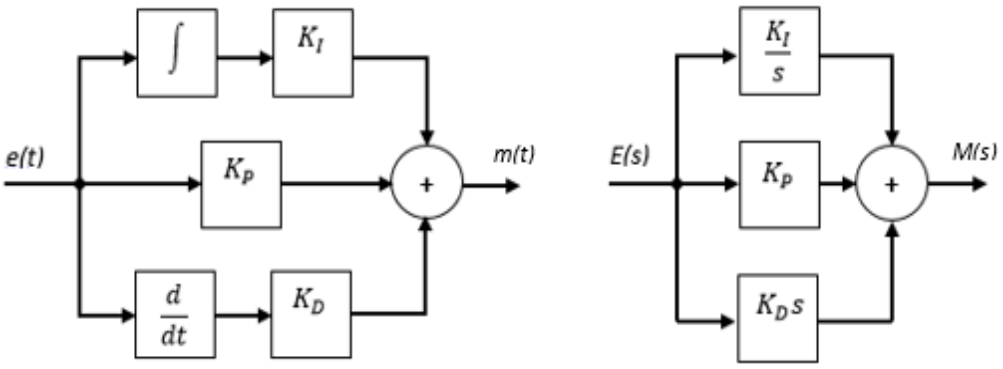
Penguat ini terbagi atas 2 bagian yaitu penguat daya dan penguat tegangan. Penguat daya dibutuhkan karena hampir semua kejadian daya dari "*error detector*" tidak cukup kuat untuk menggerakkan elemen keluaran sedangkan penguat tegangan biasanya banyak terdapat pada op-amp. Diagram blok sistem kendali industri yang terdiri dari aktuator (penggerak) dan sensor (alat pengukur) dapat dilihat pada gambar 2.3:



Gambar 2.3 Diagram Blok Sistem Kendali Industri *(Katsuhiko & Ogata, 1997)*

2.1.5 Kontrol PID (Proportional, Integral, Derivative)

Kontrol PID adalah kontrol yang terdiri dari 3 jenis cara pengaturan yang berbeda-beda yang saling dikombinasikan, yaitu P (*Proportional*), I (*Integral*) dan D (*Derivative*) (Charles L. Philips, 1997). Masing-masing memiliki parameter tertentu yang harus diset untuk dapat beroperasi dengan baik yang disebut sebagai *konstanta.* Diagram blok kontrol PID dapat dilihat pada gambar 2.4:



Gambar 2.4 Kontroler PID *(Charles L. Philips, 1997)*

2.1.5.1 Aksi Kontrol *Proportional* (P)

Untuk kontroler dengan aksi kontrol proporsional, hubungan antara masukan *u(t)* dan sinyal pembangkit kesalahan *e(t)* adalah:

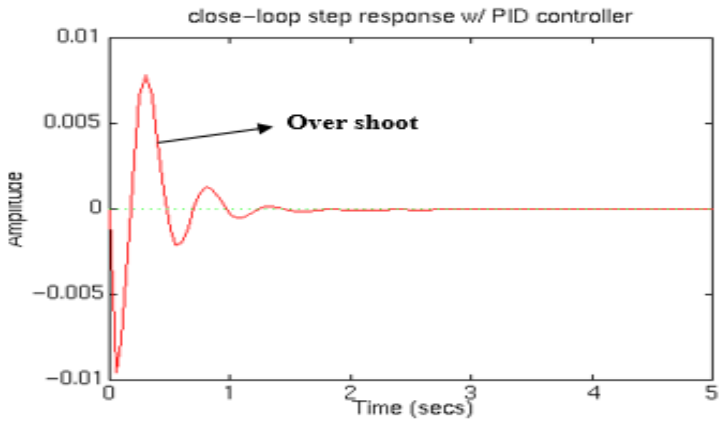
$$u(t) = K_P e(t)$$

atau dalam besaran transformasi Laplace:

$$\frac{U(s)}{E(s)} = K_P$$

dengan K_P adalah suku penguatan proporsional.

Pengendali *proportional* berfungsi untuk mempercepat proses yang dikendalikan menuju keadaan yang diinginkan (*Set Point*) (Katsuhiko & Ogata, 1997). Kecepatan proses sangat bergantung dari besarnya nilai Kp pada pengendali *proportional.* Tetapi jika nilai Kp terlalu besar maka sistem akan mengalami *over shoot* yang besar sehingga proses yang dikendalikan menjadi tidak stabil, ini dapat dilihat pada gambar 2.5:



Gambar 2.5 *Over Shoot* pada Proses *Plant* akibat Nilai Kp yang terlalu Besar

2.1.5.2 Aksi Kontrol *Integral* (I)

Pada kontroler dengan aksi kontrol integral nilai masukan kontroler *u(t)* diubah pada laju proporsional dari sinyal pembangkit kesalahan *e(t)* sehingga:

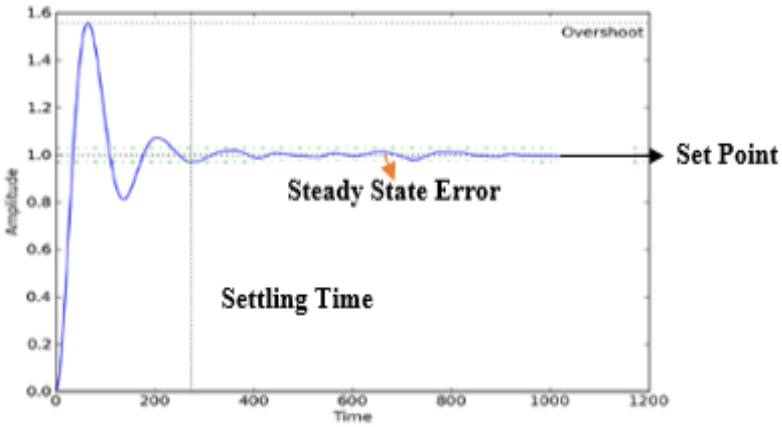
$$\frac{du(t)}{dt} = K_I e(t) \quad \boldsymbol{atau} \quad u(t) = K_I \int_0^t e(t)\, dt$$

dengan K_i adalah konstanta yang dapat diubah. Fungsi alihnya adalah:

$$\frac{U(s)}{E(s)} = \frac{K_I}{s}$$

jika nilai *e(t)* ada dua, maka nilai *u(t)* bervariasi dua kali secara cepat. Untuk pembangkit kesalahan nol, nilai *u(t)* tetap konstan. Aksi kontrol ini biasanya disebut kontrol reset (Katsuhiko & Ogata, 1997).

Pengendali *integral* berfungsi mengurangi dan menghilangkan *steady state error* yang timbul setelah respon *plant* dari pengendali *proportional* sudah stabil. Pengendali *integral* sangat optimal bekerja pada daerah di sekitar titik *set point*, yaitu antara *steady state error* dan *set point*, ini dapat dilihat pada gambar 2.6:



Gambar 2.6 Daerah Kerja Kontrol Integral pada Proses *Plant*

2.1.5.3 Aksi Kontrol *Derivative* (D)

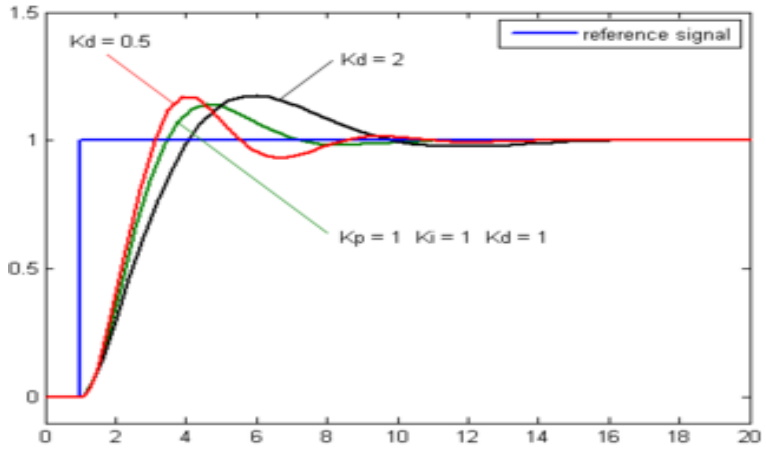
Aksi kontrol *derivative* atau turunan didefinisikan dengan persamaan berikut ini:

$$u(t) = K_D \frac{de(t)}{dt}$$

dan fungsi alihnya adalah:

$$\frac{U(s)}{E(s)} = K_D s$$

Pengendali *derivative* disebut juga kontrol laju yang berfungsi untuk mengurangi respon sistem yang terlalu berlebih yang menyebabkan *over shoot* pada proses *plant* karena nilai Kp yang terlalu besar pada pengendali *proportional* (Katsuhiko & Ogata, 1997). Pengendali *derivative* bekerja apabila terjadi perubahan *error* atau *e(t),* sehingga ketika proses yang dikendalikan sudah stabil maka pengendali *derivative* tidak akan bekerja lagi. Berikut adalah pengaruh kontrol *Derivative* pada kendali proses dapat dilihat pada gambar 2.7:



Gambar 2.7 Pengaruh Kontrol *Derivative* pada Proses *Plant*

2.1.6 Sistem Kendali Suhu Otomatis

Sistem kendali suhu otomatis adalah sistem kendali yang dapat menjaga kestabilan suhu berada pada suatu nilai suhu yang dikehendaki dengan adanya gangguan dari luar sistem (Katsuhiko & Ogata, 1997). Sistem kendali suhu otomatis ini menggunakan jenis sistem kendali lingkar tertutup atau umpan-balik (*feedback control system*) yang menggunakan kontrol PID untuk proses kontrolnya.

2.1.7 *Pulse Width Modulation* (PWM)

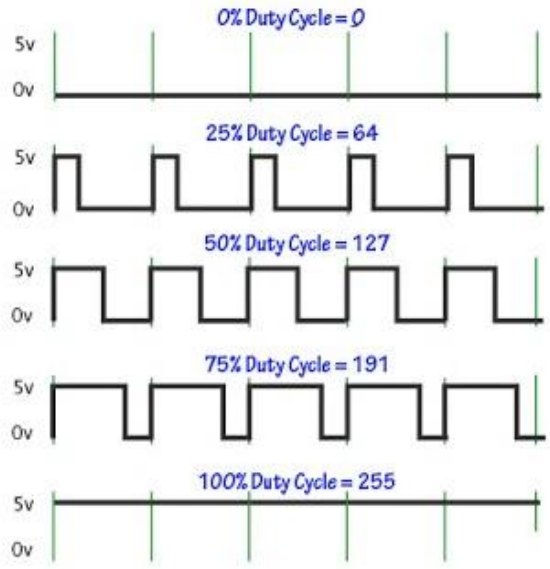
Pulse Width Modulation (PWM) secara umum adalah sebuah cara memanipulasi lebar sinyal yang dinyatakan dengan pulsa dalam satu periode untuk mendapatkan tegangan rata-rata yang berbeda. Pada penelitian ini PWM digunakan untuk menyulut rangkaian *driver heater* yang menggunakan triac agar tegangan AC 220V dapat dikontrol sehingga didapatkan besar tegangan AC yang diinginkan.

Sinyal PWM pada umumnya memiliki amplitude dan frekuensi dasar yang tetap, namun memiliki lebar pulsa yang bervariasi. Lebar pulsa PWM berbanding lurus dengan amplitude sinyal asli yang belum termodulasi. Artinya, sinyal PWM memiliki frekuensi gelombang yang tetap namun *duty cycle* bervariasi antara 0%

hingga 100%. Tegangan keluaran pada sinyal PWM dapat dicari dengan menggunakan rumus:

$$Vout = \frac{Ton}{Ton + Toff} * Vin$$

PWM merupakan salah satu teknik untuk mendapatkan sinyal analog dari sebuah piranti digital. Sebenarnya sinyal PWM dapat dibangkitkan dengan banyak cara, secara analog menggunakan IC op-amp atau secara digital. Secara analog setiap perubahan PWM-nya sangat halus, sedangkan secara digital setiap perubahan PWM dipengaruhi oleh resolusi PWM itu sendiri. Resolusi adalah jumlah variasi perubahan nilai dalam PWM tersebut. Misalkan suatu PWM memiliki resolusi 8 bit, berarti PWM ini memiliki variasi perubahan nilai sebanyak 256 variasi mulai dari 0 – 225 perubahan nilai yang mewakili *duty cycle* 0% – 100% dari keluaran PWM tersebut. Berikut adalah bentuk dari sinyal PWM sesuai dengan *duty cycle*-nya yang dapat dilihat pada gambar 2.8:



Gambar 2.8 Bentuk dari Sinyal *Pulse Width Modulation* (PWM)

2.1.8 Mikrokontroler Arduino Mega 2560

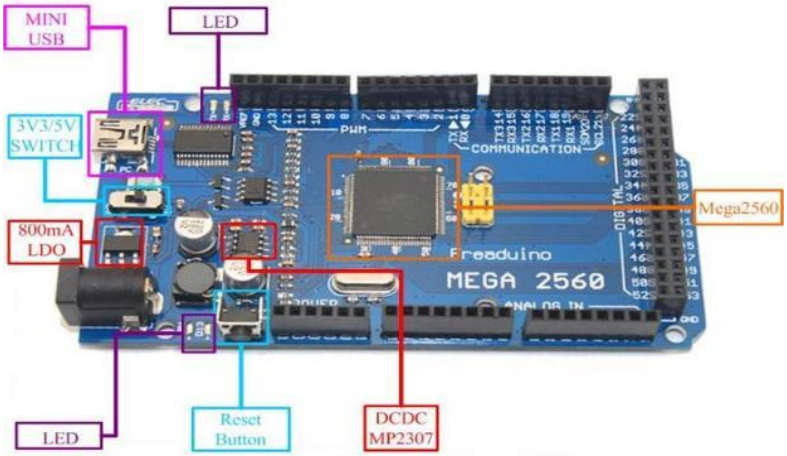
Mikrokontroler adalah sebuah sistem komputer fungsional dalam sebuah *chip*, di dalamnya terkandung inti prosesor, memori (sejumlah kecil RAM, memori program, atau keduanya), dan perlengkapan *input-output* (Budiharto & Rizal, 2007). Mikrokontroler adalah salah satu dari bagian dasar dari suatu sistem komputer. Meskipun mempunyai bentuk yang jauh lebih kecil dari suatu komputer pribadi dan komputer *mainframe*, mikokontroler dibangun dari elemen-elemen dasar yang sama. Secara sederhana, komputer akan menghasilkan *output* spesifik berdasarkan *input*an yang diterima sesuai dengan program yang dikerjakan. Seperti umumnya komputer, mikrokontroler adalah alat yang mengerjakan instruksi-instruksi yang diberikan kepadanya. Artinya, bagian terpenting dan utama dari suatu sistem terkomputerisasi adalah program itu sendiri yang dibuat oleh seorang *programmer*.

Arduino adalah kit elektronik atau papan rangkaian elektronik *open source* yang di dalamnya terdapat komponen utama, yaitu sebuah *chip* mikrokontroler dengan jenis AVR dari perusahaan Atmel (Arduino, 2015). Secara umum, Arduino terdiri dari dua bagian, yaitu:

1. Perangkat keras berupa papan *input*/*output* (I/O) yang *open source.*
2. Perangkat lunak Arduino yang juga *open source*, meliputi perangkat lunak Arduino IDE untuk menulis program dan *driver* untuk koneksi dengan komputer.

Arduino Mega 2560 adalah papan mikrokontroler berbasis ATmega2560 yang mempunyai 54 pin digital *input*/*output*, di mana 14 pin dapat digunakan sebagai *output* PWM, 16 analog *input*, 4 UARTs (*hardware serial ports*), 16 MHz

crystal oscillator, sambungan USB, *power jack*, ICSP *header*, dan tombol *reset* (Mellis, 2011). Board ini juga menggunakan daya yang terhubung ke komputer dengan kabel USB atau daya eksternal dengan adaptor AC-DC atau baterai. Arduino mega compatibel dengan *shield* yang didesain untuk Arduino Deumilanove or Diecimila. Bentuk fisik Mikrokontroler Arduino Mega 2560 dapat dilihat pada gambar 2.9:



Gambar 2.9 Arduino Mega 2560 *(Syahwil, 2013)*

2.1.9 Termometer Inframerah MLX90614

Termometer inframerah adalah sensor suhu yang menawarkan kemampuan untuk mendeteksi suhu secara optik selama objek diamati. Radiasi energi sinar inframerah diukur dan disajikan dalam bentuk suhu (Wikipedia, 2014). Termometer inframerah menawarkan metode pengukuran suhu yang cepat dan akurat dengan objek dari kejauhan dan tanpa disentuh.

MLX90614 adalah termometer inframerah untuk pengukuran suhu atau sebagai sensor pendeteksi suhu secara tak sentuh (Red, 2006). Terdapat dua buah *chip* inframerah pendeteksi suhu yang sensitif dan pengkondisian sinyal ASIC yang

terintegrasi dengan TO-39. MLX90614 memiliki penguat *noise* yang rendah. 17-bit ADC dan unit DSP yang kuat sehingga memiliki akurasi yang tinggi dalam pengukuran.

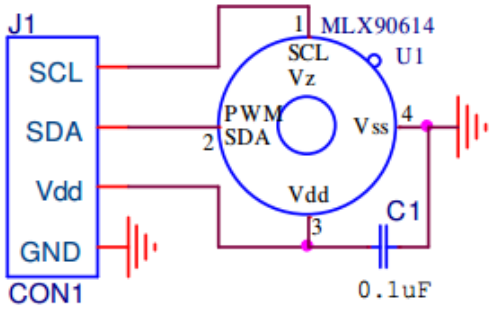
Termometer keluaran pabrik dikalibrasi dengan SMBus keluaran digital yang memberikan akses penuh ke suhu yang diukur pada rentang temperatur yang lengkap (s) dengan resolusi $0{,}02^0$C. Pengguna dapat mengkonfigurasi *output* digital menjadi PWM. Sebagai standar, 10-bit PWM dikonfigurasi untuk mengirimkan data suhu yang diukur dalam kisaran -20^0C sampai dengan 120^0C dengan resolusi *output* $0{,}14^0$C.

Ada dua cara untuk melakukan *interfacing* atau antarmuka dengan sensor MLX90614, yaitu melalui komunikasi PWM atau SMBus (TWI atau I2C). pada penelitian ini akan menggunakan metode SMBus (I2C) untuk antarmuka sensor pada Arduino. Kelebihan menggunakan metode I2C adalah kita dapat menghubungkan sampai 127 perangkat sensor pada dua buah pin yang digunakan pada Arduino sehingga akan dapat menggunakan lebih banyak sensor.

Dengan menggunakan sensor MLX90614 ini, kita dapat mengukur temperatur dari -95 sampai 720^0F (-70 sampai $382{,}2^0$C) dengan resolusi 17 bit. Hal ini artinya, sama dengan 128 kali lebih teliti dibandingkan dengan kemampuan ADC Arduino yang hanya memliki resolusi 10 bit dan ini berarti sensor MLX90614 dapat membedakan antara suhu 25^0C dengan suhu $25{,}2^0$C tanpa melakukan kontak dengan objek. Dengan resolusi sensor sampai 17 bit, maka kita akan mendapatkan resolusi sampai 0.0034^0C.

Pada penelitian ini alasan penggunaan termometer inframerah pada sistem kendali suhu adalah kebutuhan menghindari kontaminasi terhadap objek bahan

yang akan diukur, yaitu air susu. Skema rangkaian elektronik dan bentuk fisik dari sensor termometer inframerah MLX90614 yang dapat dilihat pada gambar 2.10 dan 2.11:



Gambar 2.10 Skema Rangkaian Elektronik MLX90614 *(Red, 2006)*

Gambar 2.11 Bentuk Fisik dari MLX90614 *(Red, 2006)*

2.1.10 Pemanas (*Heater*)

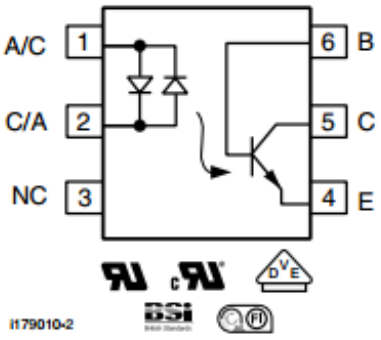
Pemanas atau *heater* adalah sebuah alat pemanas yang biasanya terbuat dari logam yang berupa lempengan, silinder pejal maupun berupa kawat pejal yang dibentuk menjadi spiral, sedangkan *hotplate* adalah sebuah pemanas yang berupa piringan yang di dalam priringan tersebut terdapat elemen pemanas yang bisa berupa logam *nichrome*, *tungsten* atau lainnya, tetapi seringkali digunakan sebagai pengganti salah satu pembakar dari berbagai oven atau bagian atas dari kompor masak. *Hotplate* atau piringan panas ini biasanya sering digunakan untuk

memanaskan makanan. Bentuk fisik pemanas (*heater*) dapat dilihat pada gambar 2.12:

Gambar 2.12 *Hotplate* atau Pemanas Makanan

2.1.11 H11AA2 (Opto-Transistor)

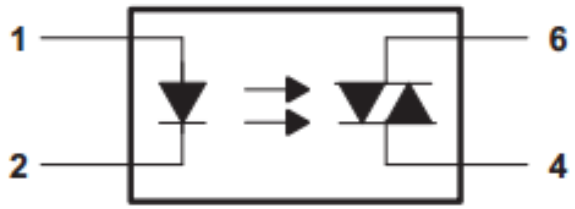
H11AA2 adalah *optoisolator* yang terdiri dari dua buah LED inframerah terbalik secara paralel dan sebuah *Photo* Transistor NPN. H11AA2 memungkinkan pemberian input tegangan secara *bi-directional* pada dua buah LED inframerah sehingga dapat digunakan sebagai aplikasi yang memerlukan deteksi atau pemantauan sinyal AC (Vishay Semiconductors, 2011). Berikut adalah rangkaian terpadu dari H11AA2 yang dapat dilihat pada gambar 2.13:



Gambar 2.13 Optoisolator H11AA2 *(Vishay Semiconductors, 2011)*

2.1.12 MOC3020 (*Opto-Triac*)

MOC3020 adalah komponen yang terdiri dari GaAs infrared pemancar dioda optik berpasangan dengan monolilitik silikon detektor yang menunjukkan fungsi sebagai pembangkit komponen triac. MOC3020 didesain untuk digunakan bersama dengan komponen TRIAC dalam antarmuka sistem logika bertegangan rendah ke peralatan bertegangan tinggi, seperti motor AC, pemanas, relai, dan lain sebagainya (Texas Instruments, 1998). Berikut adalah rangkaian terpadu dari MOC3020 yang dapat dilihat pada gambar 2.14:

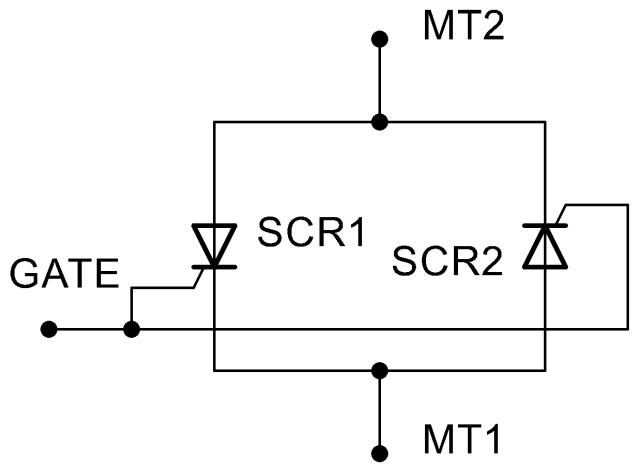


Gambar 2.14 Rangkaian Terpadu MOC3020 *(Texas Instruments, 1998)*

Pada rangkaian terpadu MOC3020 terdiri dari optoisolator dari pasangan *transmitter* dan *receiver* optik. LED sebagai *transmitter* dan *Photo Triac* sebagai *receiver*-nya. Fungsi utama dari MOC3020 ini adalah sebagai saklar optik yang tidak menimbulkan loncatan listrik (bunga api) seperti relai.

2.1.13 TRIAC

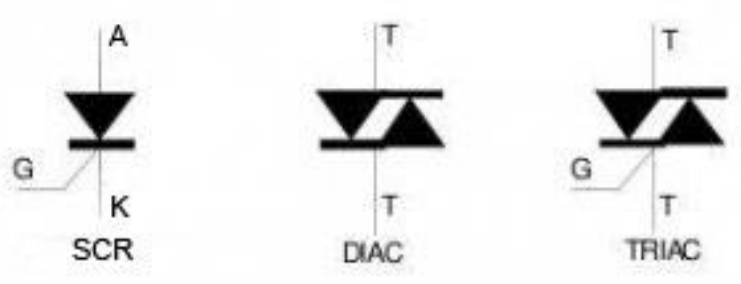
Triac adalah komponen elektronika yang digunakan untuk mengontrol daya pada listrik bolak-balik atau AC (Petruzella, 2001). Triac merupakan komponen elektronika yang terbentuk dari dua buah SCR yang dipasang terbalik secara parallel dengan kaki gate yang disatukan. Berikut adalah rangkaian ekivalen Triac yang dibentuk dengan 2 buah SCR yang dapat dilihat pada gambar 2.15:



Gambar 2.15 Rangkaian Ekivalen TRIAC dengan Dua Buah SCR

Perbedaan antara Triac dengan SCR adalah Triac dapat melewatkan arus bolak-balik jika gerbang *gate* dipicu sedangkan SCR hanya dapat melewatkan arus searah jika gerbang *gate* dipicu.

Perbedaan antara Triac dengan Diac adalah dari cara pengontrolannya, jika pada Triac gerbang *gate* perlu dipicu untuk dapat melewatkan arus bolak-balik, tetapi pada Diac, Diac harus diberikan tegangan *breakdown* tertentu pada salah satu polaritasnya barulah Diac dapat menghantarkan arus bolak-balik tersebut. Symbol SCR, DIAC, dan TRIAC dapat dilihat pada gambar 2.16:



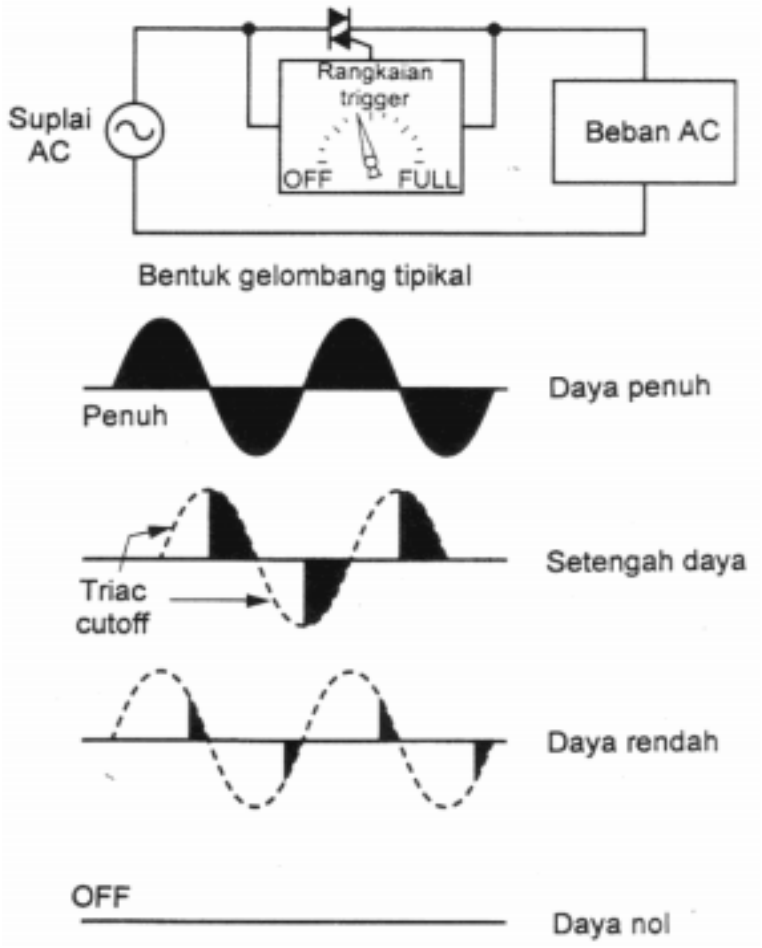
Gambar 2.16 Simbol SCR, DIAC, dan DIAC

2.1.13.1 Prinsip Kerja TRIAC

Triac dibuat untuk menyediakan cara agar kontrol daya pada listrik bolak-balik (AC) ditingkatkan (Petruzella, 2001). Triac beroperasi sebagai dua SCR dalam satu bungkus yang dipasang terbalik secara parallel. Triac memiliki kemampuan untuk mengontrol jumlah arus beban yang besar dengan jumlah arus gerbang yang kecil. Berikut adalah empat kemungkinan mode pentriggeran pada Triac sehubungan dengan MT1, antara lain:

1. MT2 adalah positif dan gerbang positif
2. MT2 adalah positif dan gerbang negatif
3. MT2 adalah negatif dan gerbang positif
4. MT2 adalah negatif dan gerbang negatif

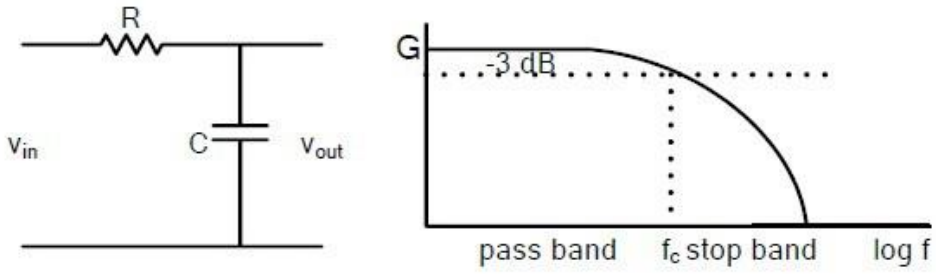
Triac dapat digunakan untuk merubah arus AC rata-rata pada beban AC seperti yang terlihat pada Gambar 2.17. Pada rangkaian penggerak (*driver*) pemanas air, perubahan arus pada pemanas air akan merubah jumlah panas yang dihasilkan oleh pemanas air sehingga suhu atau *temperature* dapat dikendalikan. Berikut adalah cara mengendalikan arus AC dengan menggunakan triac yang dapat dilihat pada gambar 2.17:



Gambar 2.17 Aplikasi Triac Sebagai Kontrol Daya Listrik AC *(Petruzella, 2001)*

2.1.14 *Low Pass Filter*

Low Pass Filter (LPF) atau filter lolos bawah adalah filter yang hanya melewatkan sinyal dengan frekuensi yang lebih rendah dari frekuensi *cut-off* (f_c) (Purnama, 2015). Pada filter LPF yang ideal sinyal dengan frekuensi di atas frekuensi *cut-off (*f_c) tidak akan dilewatkan sama sekali (tegangan = 0 volt). Rangkaian *low pass filter* RC merupakan jenis filter pasif, dengan respon frekuensi yang ditentukan oleh konfigurasi R dan C yang digunakan. Rangkaian dasar LPF dan grafik respon frekuensi sebagai berikut:



Gambar 2.18 Rangkaian Dasar dan Grafik Frekuensi *Low Pass Filter*

Frekuensi *cut-off* (f_c) dari filter pasif lolos bawah (*Low Pass Filter*) dengan RC dapt dituliskan dalam persamaan berikut:

$$f_c = \frac{1}{2\pi RC}$$

Rangkaian filter pasif LPF RC pada gambar 2.18 terlihat seperti pembagi tegangan menggunakan R, di mana pada filter LPF RC ini tegangan *output* diambil pada titik pertemuan RC. Tegangan *output* (Vout) filter pasif LPF seperti terlihat pada rangkaian di atas dapat diekspresikan dalam persamaan matematis sebagai berikut:

$$Vout = \frac{{}^{1}/_{j\omega C}}{{}^{1}/_{j\omega C} + R} * Vin$$

Besarnya penguatan tegangan (G) pada filter pasif yang ideal maksimum adalah 1 atau 0 dB yang hanya terjadi pada frekuensi sinyal input dibawah frekuensi *cut-off* (f_c). Penguatan tegangan (G) filter LPF RC pasif dapat dituliskan dalam persamaan matematis sebagai berikut:

$$G = \left|\frac{Vout}{Vin}\right|$$

Dan penguatan tegngan (G) LPF RC dapat dituliskan dalam satuan dB sebagai berikut:

$$G = 20\log\frac{Vout}{Vin} = 20\log\frac{1}{1+\omega^2C^2R^2}$$

Pada filter lolos bawah (*Low Pass Filter*) terdapat karakteristik mendasar sebagai berikut:

1. Pada saat frekuensi sinyal *input* lebih rendah dari frekuensi *cut-off* (f_c) ($f_{in} << f_c$) maka penguatan tegangan atau *gain* (G) = 1 tau G = 0 dB.
2. Pada saat frekuensi sinyal *input* sama dengan frekuensi *cut-off* (f_c) ($f_{in} = f_c$) maka $\omega = 1/RC$ sehingga penguatan tegangan atau *gain* (G) menjadi -3 dB atau terjadi pelemahan tegangan sebesar 3 dB.
3. Pada saat frekuensi sinyal *input* lebih tinggi dari frekuensi *cut-off* (f_c) ($f_{in} >> f_c$) maka besarnya penguatan tegangan (G) = $1/\omega RC$ atau G = $-20 \log \omega RC$.

2.1.15 Telemetri

Telemetri (sejenis dengan telematika) adalah sebuah teknologi yang memungkinkan pengukuran jarak jauh dan pelaporan informasi kepada perancang atau operator sistem (No, 1998). Kata telemetri berasal dari akar bahasa Yunani *tele* yang berarti jarak jauh, dan *metron* yang berarti pengukuran (Wikpedia, 2014). Sistem yang membutuhkan instruksi atau data yang dikirim kepada mereka untuk mengoperasikan membutuhkan lawan dari telemetri adalah telekomando. Telemetri merujuk pada komunikasi nirkabel (contohnya menggunakan sistem radio untuk mengimplementasikan hubungan data), tapi juga merujuk pada data yang dikirimkan melalui media lain, seperti telepon atau jaringan komputer atau melalui kabel optik atau ketika membuat robot kita dapat menggunakan satu kabel. Seperti

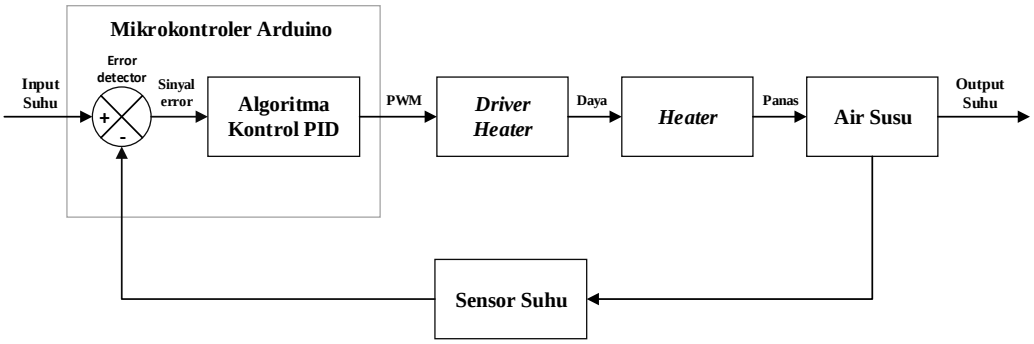
telekomunikasi lainnya. Standar internasional ditetapkan untuk peralatan dan peranti lunak telemetri adalah CCSDS (*Consultative Committee for Standard Data Services*) dan IRIG (*The Standard for Digital Flight Data Recording*).

2.1.15.1 Bluetooth

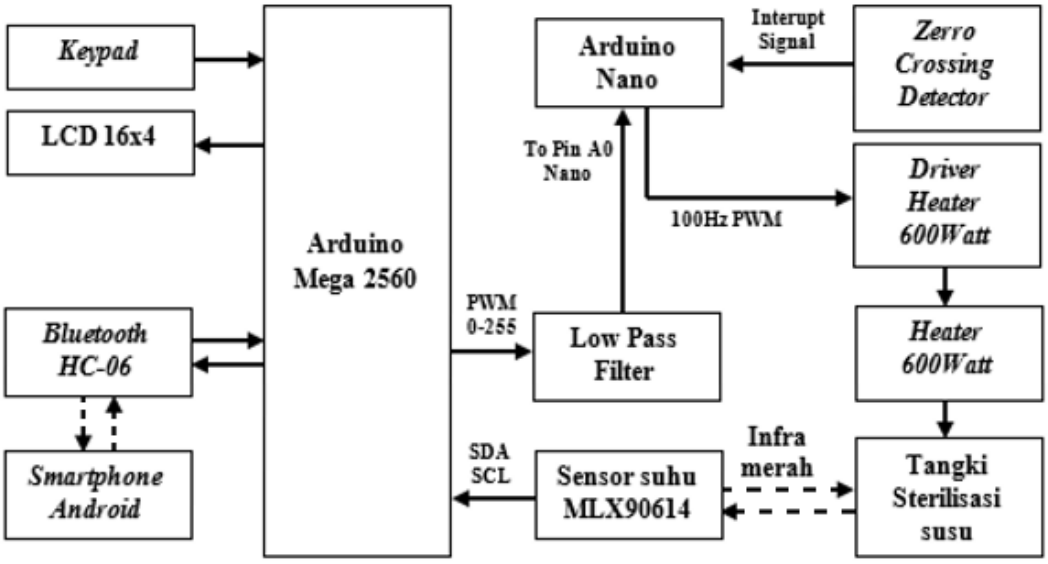
Bluetooth adalah spesifikasi industri untuk jaringan kawasan pribadi (*personal area* networks) tanpa kabel (Davis, 2002). *Bluetooth* menghubungkan dan dapat dipakai untuk melakukan tukar-menukar informasi di antara peralatan-peralatan. Spesifikasi dari peralatan *Bluetooth* beroperasi dalam pita frekuensi 2.4 GHz dengan menggunakan *frequency hoping traceiver* yang mampu menyediakan layanan komunikasi data dan suara secara *real time* antara *host to host Bluetooth* dengan jarak terbatas. Kelemahan teknoogi ini adalah jangkauannya yang pendek dan kemampuan transfer data yang rendah.

2.2 Kerangka Berpikir

2.2.1 Diagram Blok Sistem Kendali

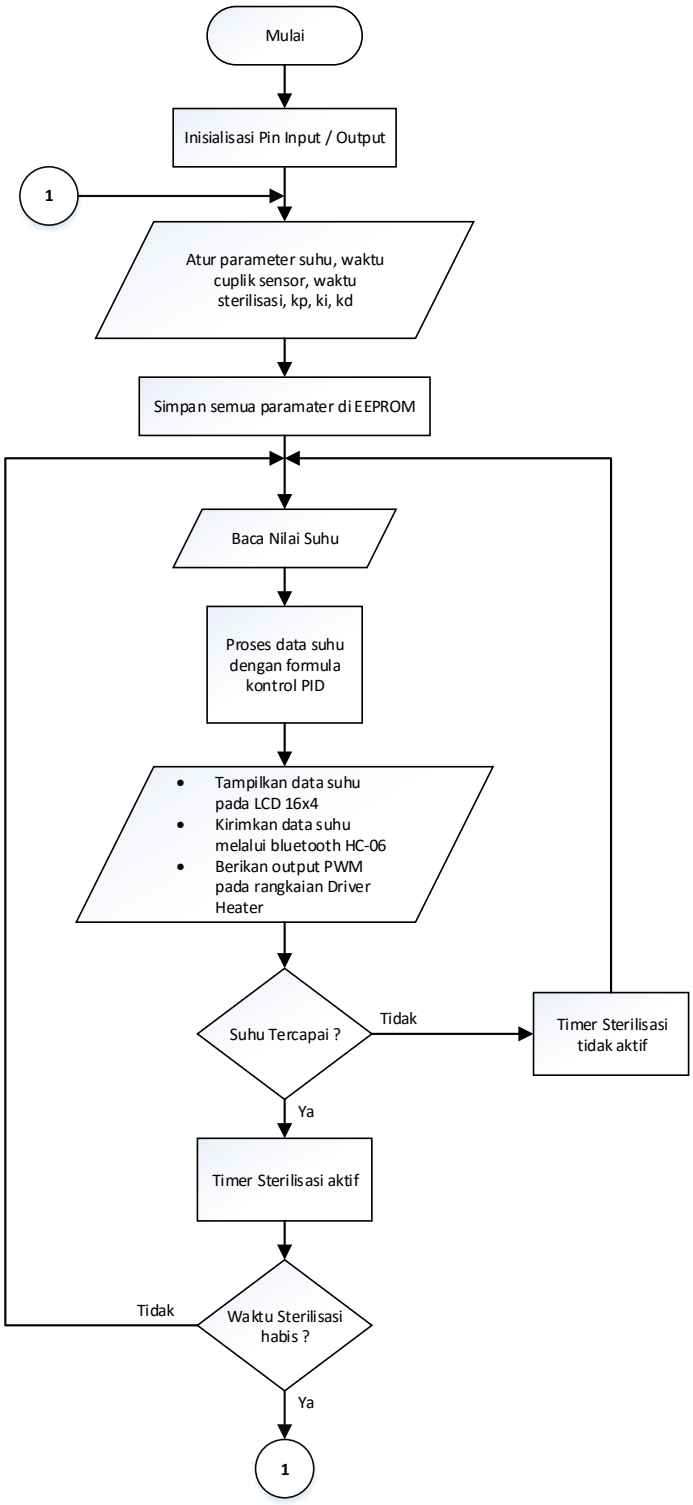


Gambar 2.19 Diagram Blok Sistem Kendali Suhu *Loop*-tertutup



Gambar 2.20 Diagram Blok Sistem Kendali Suhu Proses Sterilisasi Susu

2.2.2 Diagram Alir Sistem Kendali Suhu untuk Proses Sterilisasi Susu



Gambar 2.21 Diagram Alir Sistem Kendali Suhu untuk Proses Sterilisasi Susu

2.3 Hipotesis Penelitian

Sistem Kendali Suhu Otomatis untuk Proses Sterilisasi Susu dengan Metode Konvensional Menggunakan Kontrol PID Berbasis Arduino Mega 2560 dengan Telemetri dapat menjaga kestabilan suhu pada saat proses sterilisasi susu dilakukan, yaitu pada suhu 120^0C selama 15 menit dan proses tersebut dapat dipantau dan dikontrol dari jarak jauh (telemetri) dengan menggunakan *smartphone* Android melalui koneksi *Bluetooth*.

BAB III

METODOLOGI PENELITIAN

3.1 Tujuan Penelitian

Tujuan dari penelitian ini adalah merancang dan membuat sistem kendali suhu otomatis yang dapat menjaga kestabilan suhu pada proses sterilisasi susu dengan metode konvensional dan proses tersebut dapat dipantau dan dikontrol dari jarak jauh (telemetri) dengan menggunakan *smartphone* Android melalui koneksi *Bluetooth.*

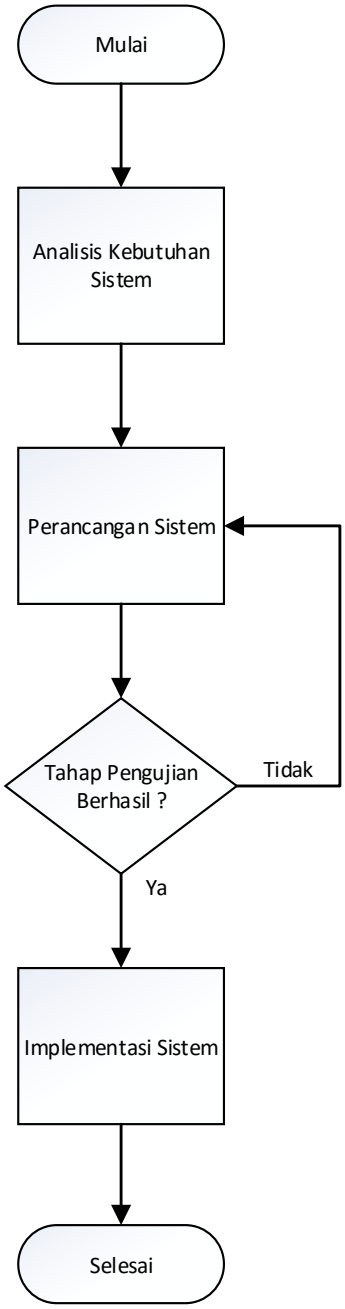
3.2 Tempat Dan Waktu Penelitian

Penelitian dilaksanakan di Gedung Teknik Elektro Universitas Negeri Jakarta pada bulan Juli 2015 – Desember 2015. Waktu tersebut cukup efektif untuk melakukan penelitian.

3.3 Metode Penelitian

Metode Penelitian dapat diartikan sebagai langkah-langkah penelitian suatu produk yang akan dikembangkan atau dibuat. Metode yang digunakan untuk merancang bangun *Sistem Kendali Suhu Otomatis pada Proses Sterilisasi Susu dengan Metode Konvensional Menggunakan Kontrol PID Berbasis Arduino Mega 2560 dengan Telemetri* adalah metode penelitian dan pengembangan (*Research and Development*) sebuah produk untuk menghasilkan sebuah sistem kendali suhu otomatis yang dapat menjaga kestabilan suhu untuk proses sterilisasi susu yang dapat dipantau dan dikontrol dari jarak jauh (telemetri) dengan menggunakan *smartphone* Android.

Berikut adalah tahapan-tahapan metodologi penelitian yang akan dilakukan oleh peneliti pada penelitian ini yang dapat dilihat pada gambar 3.1:



Gambar 3.1 Tahap-Tahap Metodologi Penelitian

3.3.1 Analisis Kebutuhan Sistem

Analisa kebutuhan sistem adalah tahap di mana peneliti menentukan apa saja yang dibutuhkan oleh sistem agar dapat berjalan sesuai dengan tujuan penelitian. Untuk memenuhi tujuan tersebut peneliti memerlukan beberapa kebutuhan sistem sebagai berikut:

1. Sensor suhu yang dapat mengukur suhu pada objek secara tak sentuh yang memiliki akurasi tinggi, respon yang cepat, dan stabil.
2. Sistem kontrol suhu yang stabil.
3. Sistem kontrol yang tepat waktu.
4. Sistem kontrol yang dapat dikontrol dan dipantau dari jarak jauh (telemetri).

3.3.2 Perancangan Sistem

Perancangan sistem pada penelitian ini sangat dibutuhkan agar sistem yang ingin direalisasikan dapat berjalan seperti yang diinginkan. Berikut adalah rancangan sistem pada penelitian ini antara lain:

1. Merancang dan membuat antarmuka sensor suhu MLX90614 pada Arduino Mega 2560 dengan komunikasi I2C agar dapat mendeteksi suhu.
2. Merancang dan membuat antarmuka LCD 16x4 pada Arduino Mega 2560 agar dapat menampilkan nilai suhu yang dibaca oleh sensor suhu.
3. Merancang dan membuat antarmuka tombol-tombol navigasi dari sistem kendali pada Arduino Mega 2560 agar sistem kendali dapat dioperasikan.
4. Merancang dan membuat antarmuka *driver* pemanas air pada Arduino Mega 2560 agar sistem kendali dapat mengatur pemberian besar daya pada pemanas air secara otomatis.

5. Merancang dan membuat algoritma kontrol PID pada Arduino Mega 2560 untuk sistem kendali agar sistem kendali dapat menjaga kestabilan suhu selama proses sterilisasi dilakukan.
6. Merancang dan membuat antarmuka telemetri pada Arduino Mega 2560 dan *smartphone* Android secara nirkabel (telemetri) dengan komunikasi *Bluetooth* agar sistem kendali dapat dipantau dari jarak jauh.

3.3.3 Pengujian Sistem

Pengujian sistem adalah tahap di mana peneliti melakukan pengujian terhadap hasil rancangan sistem yang telah dibuat sebelumnya. Berikut ini adalah tahap-tahap pengujian pada penelitian ini antara lain:

1. Tahap pertama uji coba dilakukan terhadap antarmuka sensor suhu MLX90614 dan LCD 16x4 secara bersamaan. Ini untuk mengetahui apakah sensor suhu sudah dapat berkomunikasi dengan baik pada Arduino Mega 2560 atau belum?, keberhasilan ditandai dengan tampilnya data suhu pada layar LCD 16x4.
2. Tahap kedua uji coba dilakukan terhadap rangkaian *driver* pemanas air dalam mengatur besar daya yang diberikan pada pemanas air. Pengujian ini dilakukan dengan pemberian lebar pulsa PWM pada rangkaian *driver* pemanas air, setelah itu dilakukan pengukuran terhadap tegangan *output* rangkaian dengan menggunakan AVO meter, keberhasilan ditandai dengan tegangan *output* rangkaian yang berubah-ubah sesuai dengan lebar pulsa PWM yang diberikan.
3. Tahap ketiga uji coba dilakukan terhadap antarmuka tombol-tombol navigasi sistem kendali untuk menguji apakah tombol-tombol navigasi tersebut dapat

digunakan untuk mengoperasikan sistem kendali dengan benar sesuai dengan fungsi dari setiap tombol tersebut

4. Tahap keempat uji coba dilakukan terhadap antarmuka telemetri pada Arduino Mega 2560 dan *smartphone* Android. Uji coba dilakukan untuk mengetahui seberapa jauh jangakauan komunikasi *Bluetooth* di antara kedua perangkat, dengan cara mengirimkan data suhu dari Arduino Mega 2560 ke *smartphone* Android, jika berhasil maka suhu akan tampil pada layar *smartphone* Android.

3.3.4 Implementasi Sistem

Implementasi sistem adalah penerapan sistem kendali yang pada tahap sebelumnya sudah dirancang, dibuat dan diuji pada lapangan. Implementasi sistem bertujuan untuk mengetahui apakah produk yang dibuat telah mampu mencapai tujuan dari penelitian atau belum?.

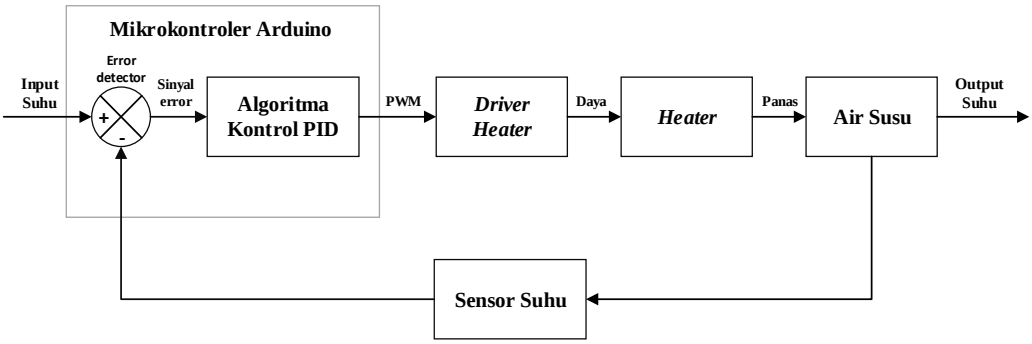
Pada penelitian ini akan dilakukan implementasi sistem sebagai berikut:

1. Melakukan *tuning* parameter PID dengan metode coba-coba (*trial & error)* pada sistem kontrol suhu sehingga didapatkan sistem kontrol suhu yang stabil untuk proses sterilisasi susu.
2. Melakukan pemantauan dan pengontrolan sistem kontrol suhu secara nirkabel (telemetri) dengan menggunakan *smartphone* Android.

3.4 Rancangan Penelitian

Rancangan penelitian merupakan suatu rencana yang komprehensif dan memiliki tujuan yang terarah dalam melakukan penelitian untuk mengasilkan karya yang diinginkan. Berikut adalah rancangan penelitian pada penelitian ini:

3.4.1 Membuat Diagram Blok Sistem Kendali



Gambar 3.2 Diagram Blok Sistem Kendali Suhu *Loop*-tertutup

Berdasarkan diagram blok sistem kendali suhu *loop*-tertutup di atas, berikut adalah fungsi dari setiap sub sistemnya antara lain:

1. *Input* suhu sebagai suhu referensi yang harus dicapai harus sistem kendali.
2. *Error detector* sebagai penghitung besar *error* yang terjadi pada *output* suhu terhadap *input* suhu.
3. Algoritma kontrol PID sebagai pengendali suhu agar tetap stabil sesuai dengan suhu referensi atau *input* suhu, *output* kontrol PID di sini adalah berupa PWM (*Pulse Width* Modulation) yang akan menggerakkan rangkaian *driver heater*berikut adalah algoritma dari kontrol PID:
   a. Mulai
   b. Atur nilai input suhu yang diinginkan (SP) dan parameter KP, KI, KD
   c. Baca nilai suhu pada air susu (PV)
   d. Hitung nilai *Error* = SP - PV
   e. Hitung aksi kontrol P = KP * (*Error*)
   f. Hitung aksi kontrol I = KI * (*Error* + *Error* yang lalu)

g. Hitung aksi kontrol D = KD * (*Error* – *Error* yang lalu)

h. Hitung PWM = P + I + D

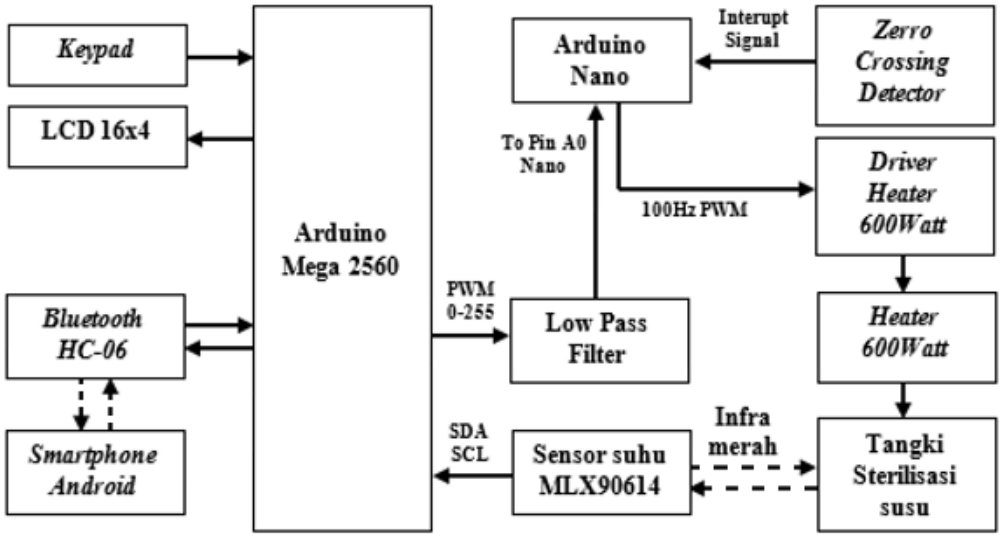
i. Cetak PWM

j. *Error* yang lalu = *Error*

k. Kembali ke langkah c

4. *Driver heater* digunakan sebagai penggerak dari *heater* atau sebagai pengatur jumlah daya yang diberikan pada *heater*, *driver heater* dikontrol dengan menggunakan PWM.
5. *Heater* sebagai pemanas untuk memanaskan air susu sesuai dengan daya yang diberikan oleh *driver heater*
6. Air susu merupakan *plant* yang dikendalikan suhunya agar tetap stabil.
7. Sensor suhu sebagai pendeteksi pada air susu.

Berikut adalah diagram blok sistem kontrol suhu untuk proses sterilisasi susu secara keseluruhan yang dapat dilihat pada gambar 3.3:



Gambar 3.3 Diagram Blok Sistem Kontrol Suhu untuk Proses Sterilisasi Susu

Berdasarkan diagram blok pada Gambar 3.3, berikut adalah fungsi dari bagian-bagian sistem kendali tersebut antara lain:

1. *Keypad* sebagai masukan bagi Arduino Mega 2560 untuk melalukan navigasi dalam melakukan kalibrasi parameter-parameter yang dibutuhkan oleh sistem kendali
2. LCD 16x4 akan menampilkan seluruh informasi-informasi penting dari sistem kendali yang berasal Arduino Mega 2560
3. *Bluetooth* HC-06 sebagai perangkat telemetri sistem kendali agar sistem kendali dapat berkomunikasi dengan Smartphone Android secara nirkabel melalui modus serial
4. Arduino Mega 2560 sebagai pusat kendali dari sistem kendali yang melakukan seluruh pemrosesan data *input-output* dari sistem kendali seperti pemrosesan data pembacaan suhu oleh sensor suhu, pemrosesan data input dari *keypad*, pemrosesan data *output* untuk LCD 16x4 dan lain sebagainya
5. *Low Pass Filter* akan memfilter frekuensi sinyal PWM yang berasal dari keluaran Arduino Mega untuk dikonversi menjadi tegangan analog tanpa frekuensi yang akan dikirimkan ke Arduino Nano untuk diproses
6. Arduino Nano akan memproses tegangan analog dari rangkaian *Low Pass Filter* untuk dijadikan acuan berapa besar sinyal PWM dengan frekuensi 100Hz yang akan dikrimkan ke rangkaian *driver heater*
7. MLX90614 sebagai sensor pengukur suhu pada air susu secara tak sentuh dengan komunikasi SDA dan SCL pada Arduino Mega 2560
8. Rangkaian *Zero Crossing Detector* akan mendeteksi kapan terjadinya tegangan 0V pada sumber tegangan AC220V dengan frekuensi 50Hz, yang kemudian

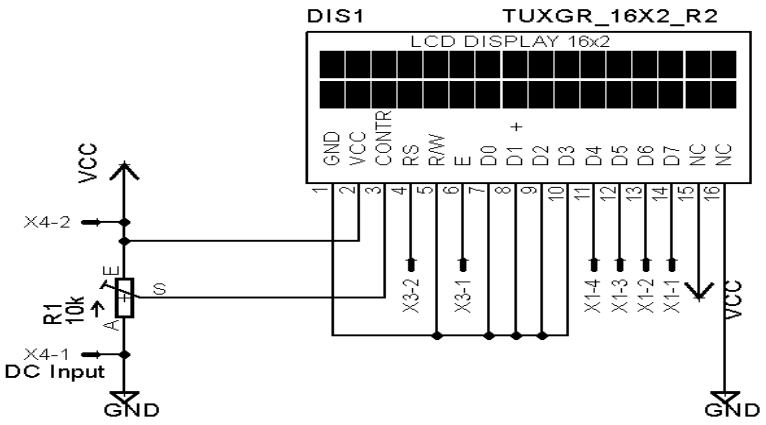
akan menghasilkan sinyal interupsi untuk Arduino Nano agar Arduino dapat memberikan sinyal PWM pada rangkaian *heater* dengan frekuensi 100Hz

9. Rangkaian d*river heater* akan memberikan berapa besar daya yang akan dikirimkan pada heater bergantung dari berapa besar sinyal PWM yang dikirimkan oleh Arduino Nano
10. *Heater* akan merubah daya listrik yang diberikan oleh rangkaian *driver heater* menjadi panas untuk memanaskan tangki sterilisasi susu.

3.4.2 Perancangan Perangkat Keras

3.4.2.1 Perancangan Rangkaian LCD

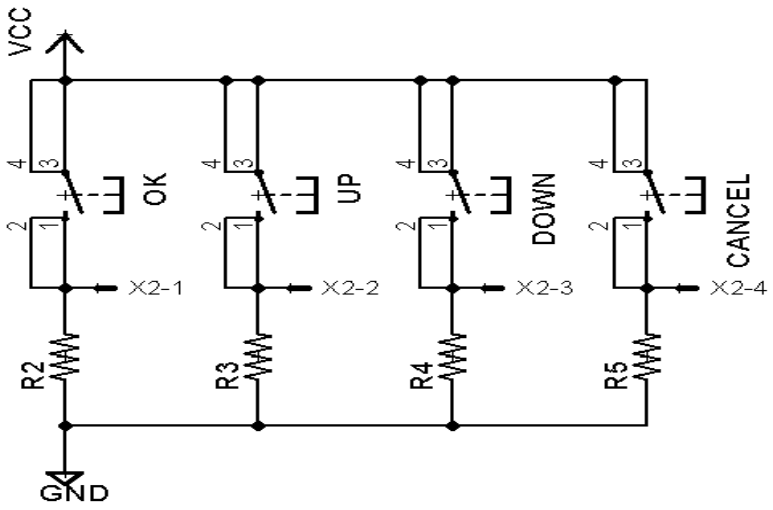
Rangkaian LCD merupakan rangkaian yang dirancang agar Arduino Mega 2560 dapat berinteraksi dengan LCD untuk menampilkan informasi-informasi dari sistem. Rangkaian LCD membutuhkan tegangan +5VDC agar dapat menyala dan 4 buah jalur data agar dapat berkomunikasi dengan Arduino Mega 2560 untuk menampilkan informasi yang terdapat pada sistem serta 1 buah komponen trimpot 10k untuk mengatur kontras dari LCD. Rangkaian LCD 16x4 dapat dilihat pada gambar 3.4:



Gambar 3.4 Rangkaian LCD

3.4.2.2 Perancangan Rangkaian *Keypad*

Rangkaian *keypad* merupakan rangkaian yang digunakan sebagai rangkaian *input* untuk Arduino Mega 2560 agar dapat digunakan untuk mengatur parameter-parameter dari sistem kendali. Rangkaian *Keypad* dilengkapi dengan resistor *pull down* dan sumber tegangan +5VDC agar dapat dioperasikan. Cara kerja rangkaian *keypad* adalah pada saat *keypad* ditekan rangkaian *keypad* akan menghasilkan sinyal keluaran berlogika *HIGH* atau (1), sebaliknya ketika rangkaian *keypad* tidak ditekan akan menghasilkan sinyal keluaran berlogika *LOW* atau (0). Rangkaian *keypad* dapat dilihat pada gambar 3.5:

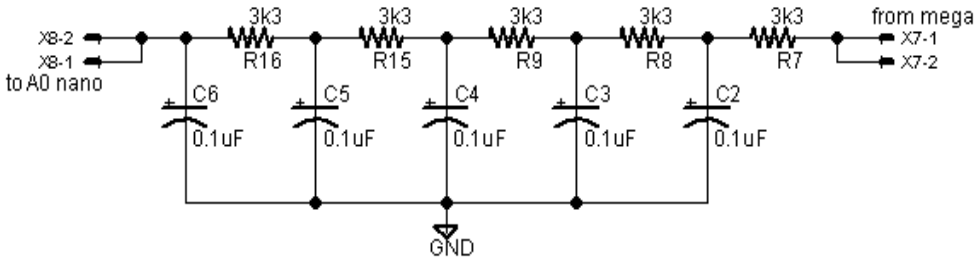


Gambar 3.5 Rangkaian *Keypad*

3.4.2.3 Perancangan Rangkaian *Low Pass Filter*

Rangkaian *low pass filter* merupakan rangkaian yang dirancang untuk menghilangkan frekuensi dari sinyal PWM yang berasal dari Arduino Mega 2560 agar didapatkan tegangan analog tanpa frekuensi. Fungsi tegangan analog dari keluaran rangkaian *low pass filter ini* adalah untuk digunakan sebagai referensi bagi Arduino Nano untuk menentukan berapa besar sinyal PWM dengan Frekuensi

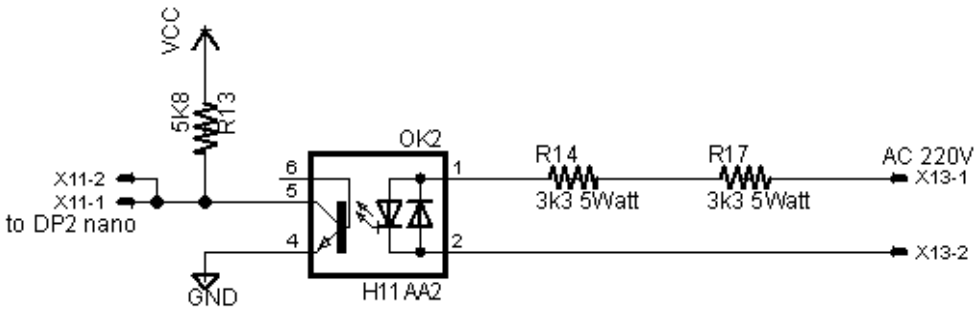
100Hz yang dikirimkan pada rangkaian *driver heater*. Rangkaian *low pass filter* dapat dilihat pada gambar 3.6:



Gambar 3.6 Rangkaian *Low Pass Filter*

3.4.2.4 Perancangan Rangkaian *Zero Crossing Detector*

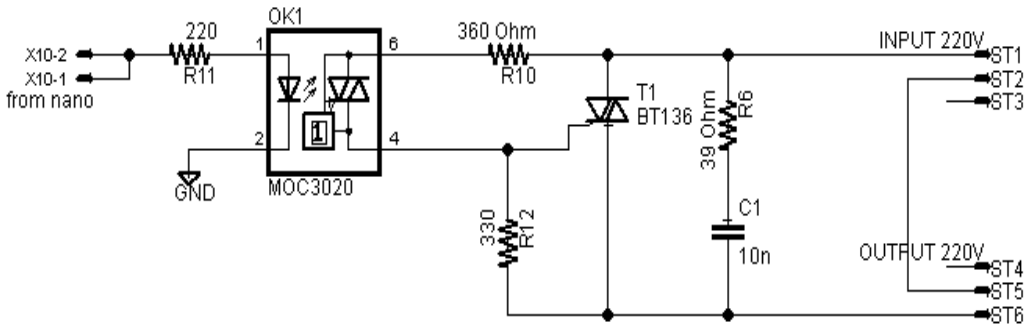
Rangkaian *zero crossing detector* merupakan rangkaian yang digunakan untuk menghasilkan sinyal interupsi bagi Arduino Nano. Rangkaian *Zero crossing detector* ini memiliki komponen utama H11AA2 yang memiliki 2 buah inframerah yang disusun terbalik secara paralel dan sebuah *phtoto* transistor di dalamnya sehingga dapat digunakan untuk mendeteksi kapan terjadinya tegangan 0V pada tegangan AC 220V. Ketika tegangan AC 220V belum mencapai 0V maka keluaran logika dari rangkaian *zero crossing detector* akan bernilai *LOW* (0), ketika tegangan AC 220V mencapai 0V maka keluaran logika dari rangkaian *zero crossing detector* akan bernilai *HIGH* (1) dan begitu seterusnya. Perubahan logika dari keluaran rangkaian *zero crossing detector* dari *LOW* (0) menjadi *HIGH* (1) akan digunakan sebagai sinyal interupsi yang mengaktifkan Arduino Nano untuk memberikan sinyal PWM dengan frekuensi 100Hz ke rangkaian *driver heater*. Rangkaian *zero crossing detector* dapat dilihat pada gambar 3.7:



Gambar 3.7 Rangkaian *Zero Crossing Detector*

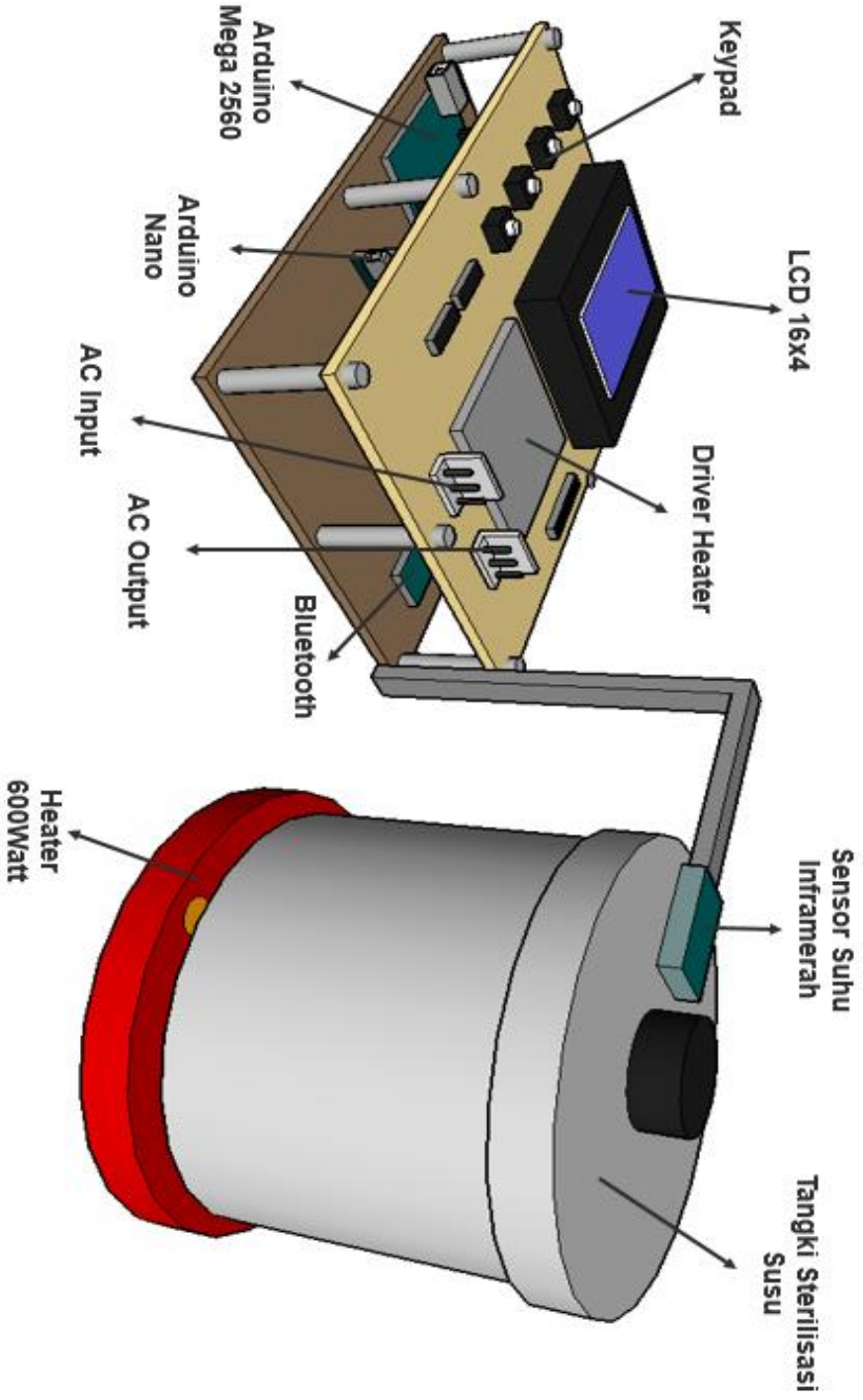
3.4.2.5 Perancangan Rangkaian *Driver Heater*

Rangkaian *driver heater* merupakan rangkaian yang digunakan sistem kendali melalui Arduino Nano agar dapat mengatur besar daya yang akan diberikan pada *heater* sehingga dapat mengatur jumlah panas yang akan diberikan pada tangki sterilisasi susu. Rangkaian *driver heater* dilengkapi dengan sebuah Opto Triac MOC3020 yang berfungsi untuk mengisolasi rangkaian kontrol bertegangan rendah terhadap rangkaian yang dikontrol dengan tegangan tinggi dan sebuah Triac BT137 untuk menguatkan arus listrik yang akan dikirim ke *heater* dengan arus maksimal sebesar 8 *Ampere.* Rangkaian *driver heater* dapat dilihat pada gambar 3.8:



Gambar 3.8 Rangkaian *Driver Heater*

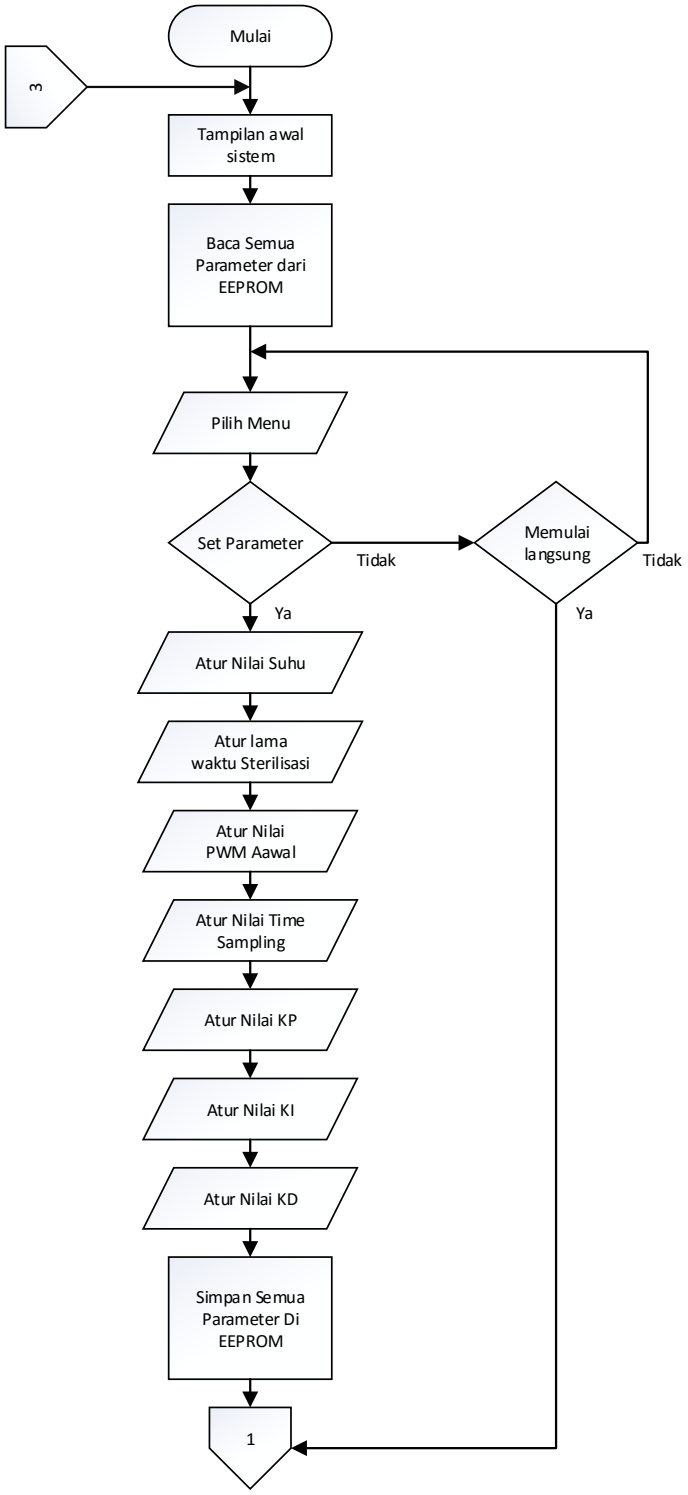
3.4.2.6 Desain Maket Sistem Kendali



Gambar 3.9 Desain Maket Sistem Kendali

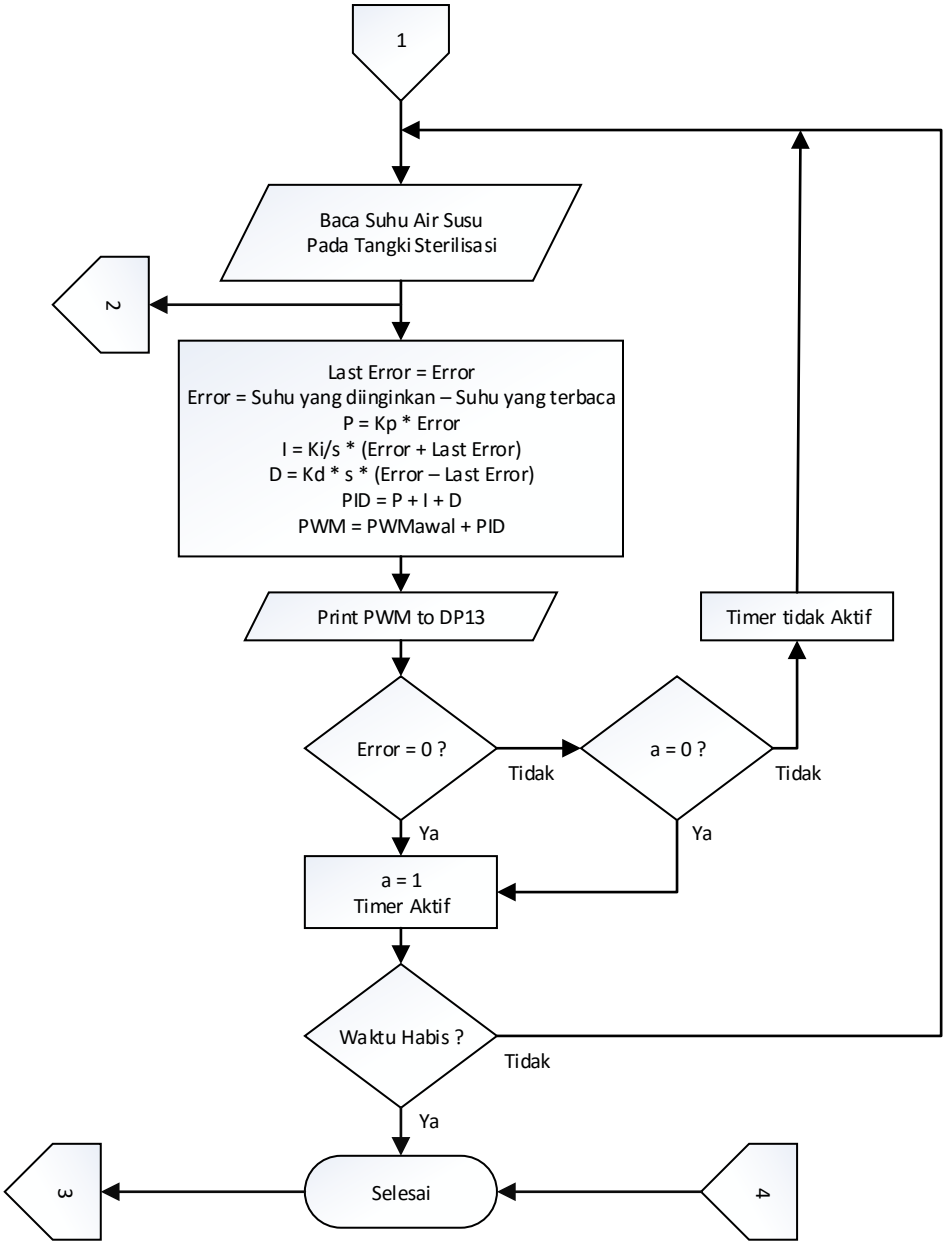
3.4.3 Perancangan Perangkat Lunak

3.4.3.1 *Flowchart* Kalibrasi Sistem Kendali Arduino pada Mega 2560



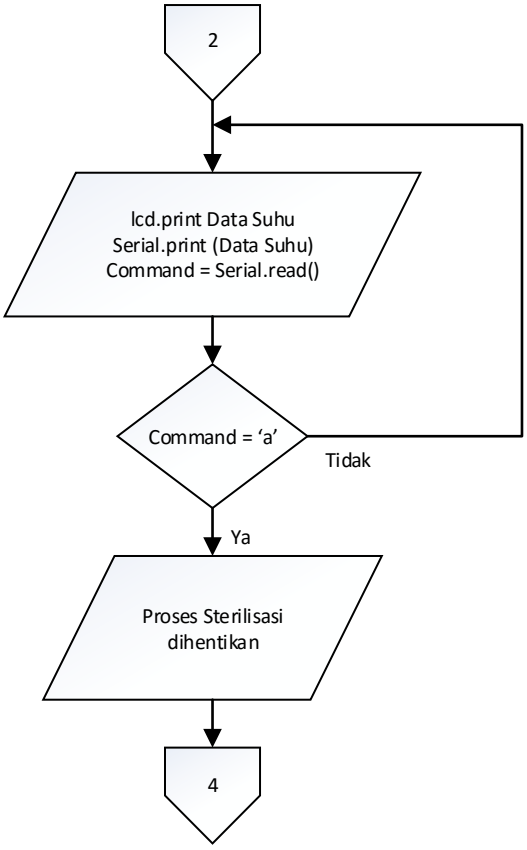
Gambar 3.10 *Flowchart* Kalibrasi Sistem Kendali pada Arduino Mega 2560

3.4.3.2 *Flowchart* Pembacaan Suhu dan Kontrol PID pada Arduino Mega 2560



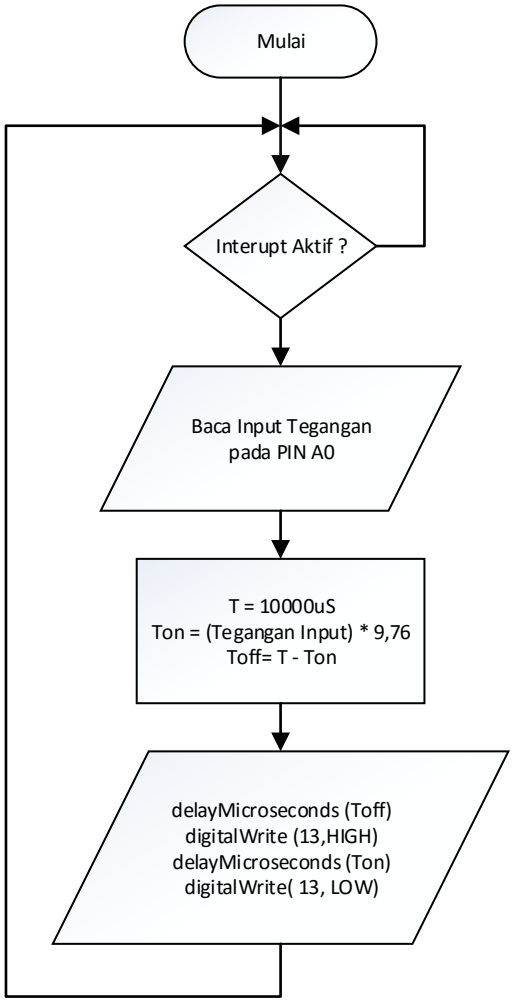
Gambar 3.11 *Flowchart* Pembacaan Suhu dan Kontrol PID pada Arduino Mega 2560

3.4.3.3 *Flowchart* Pengiriman Data Suhu ke *Bluetooth* dan LCD Arduino Mega 2560



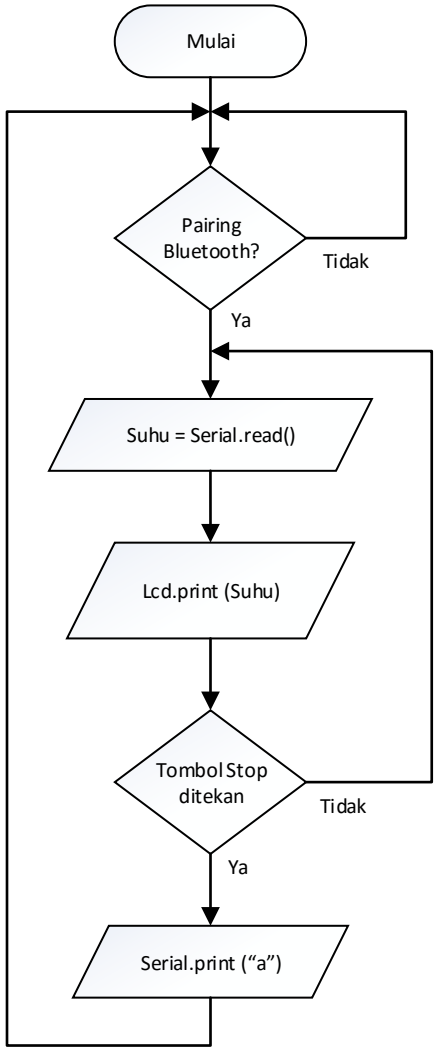
Gambar 3.12 *Flowchart* Pengiriman Data Suhu ke Bluetooth HC-06 dan LCD pada Arduino Mega 2560

3.4.3.4 *Flowchart* Pembangkit PWM 100Hz pada Arduino Nano



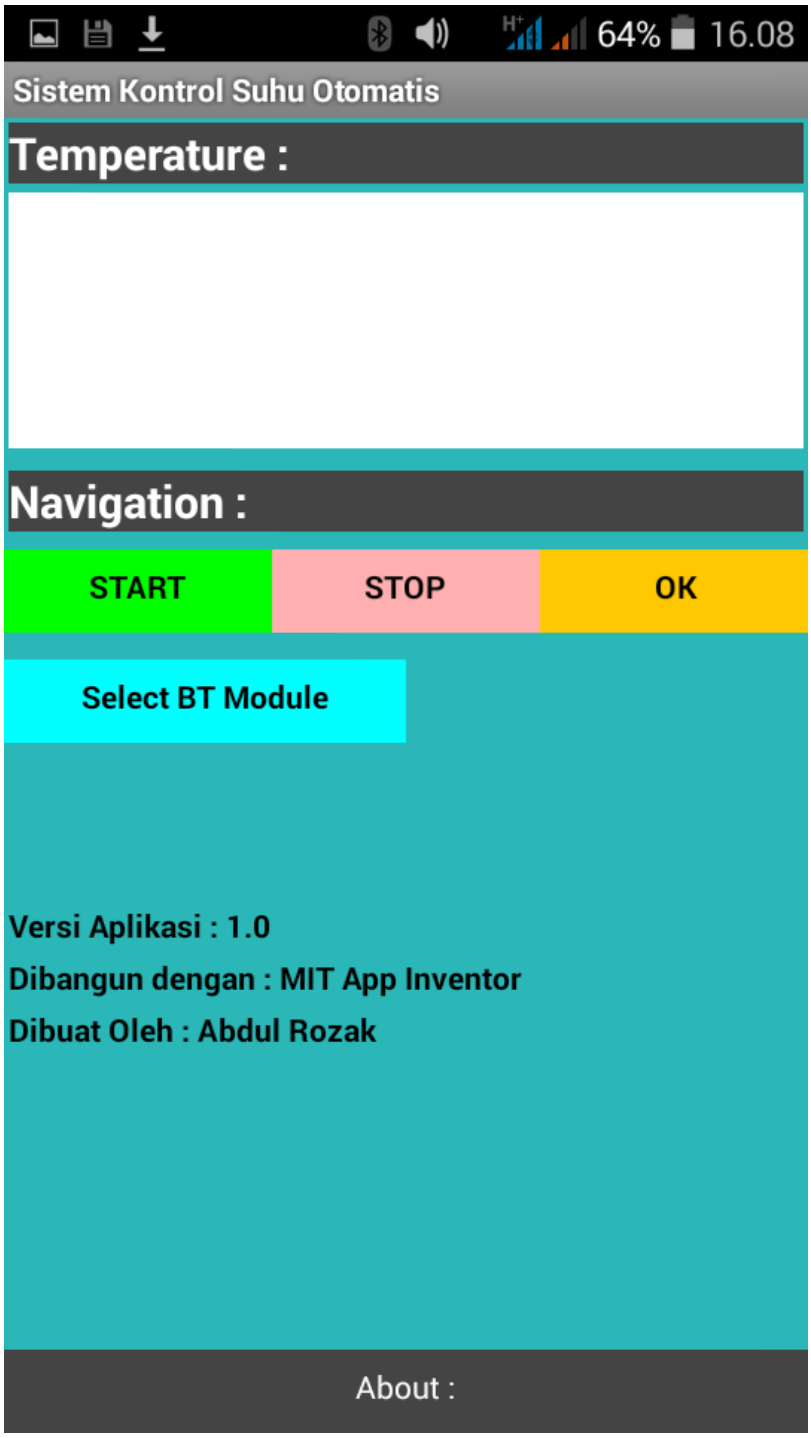
Gambar 3.13 *Flowchart* Pembangkit PWM 100Hz pada Arduino Nano

3.4.3.5 *Flowchart* Telemetri pada *Smartphone* Android



Gambar 3.14 *Flowchart* Telemetri untuk *Smartphone* Android

3.4.3.6 Desain Aplikasi untuk *Smartphone* Android



Gambar 3.15 Desain Aplikasi Telemetri untuk *Smarphone* Android

3.5 Instrumen Penelitian

Untuk memberikan hasil yang akurat dan presisi pada saat pengambilan dalata dalam penelitian, peneliti menggunakan beberapa instrument penelitian sebagai berikut:

1. Sistem komputer yang digunakan dalam penelitian dengan spesifikasi sebagai berikut:
    a. *Processor* AMD C70 1 GHz *up to* 1,33 GHz
    b. RAM 4 GB DDR3
    c. *Hardisk* 512 GB
    d. AMD Radeon HD 7290 Grapichs
    e. Sistem Operasi Microsoft Windows 10 Pro 64 bit
2. Software Pendukung yang digunakan dalam penelitian sebagai berikut:
    a. *Software* Arduino IDE 1.6.5 yang digunakan untuk membuat program sistem kendali berbasis Arduino.
    b. *Software* App Inventor 2 yang digunakan untuk membuat program aplikasi pembacaan suhu menggunakan *Smartphone* Android.
    c. *Software* EAGLE Layout Editor 6.4.0 yang digunakan untuk membuat rancangan rangkaian dan PCB untuk sistem kendali.
    d. Microsoft Word 2016 yang digunakan untuk penulisan.
    e. Microsoft Visio 2016 yang digunakan untuk pembuatan *Flowchart* program sistem kendali.

3.6 Prosedur Penelitian

Prosedur Penelitian merupakan tahap-tahap yang akan dilakukan oleh peneliti agar mendapatkan hasil yang optimal sesuai dengan yang diharapkan. Berikut adalah tahap-tahap yang akan dilakukan peneliti, antara lain:

1. Membuat diagram blok sistem kendali.
2. Membuat rancangan perangkat keras sistem kendali.
3. Membuat perangkat keras sistem kendali.
4. Membuat rancangan perangkat lunak sistem kendali.
5. Membuat perangkat lunak sistem kendali.
6. Uji coba perangkat keras sistem kendali.
7. Uji coba perangkat lunak sistem kendali.
8. Uji coba sistem kendali secara keseluruhan.
9. Implementasi Sistem.

3.7 Teknik Analisis Data

Teknik analisis data merupakan kriteria pengujian yang dilakukan peneliti untuk mendapatkan data yang diperlukan pada keseluruh sistem yang dibuat. Kriteria pengujian dilakukan peneliti untuk menyatakan bahwa sistem yang telah dibuat dinyatakan berhasil atau gagal. Berikut kriteria pengujian pada penelitian *Sistem Kendali Suhu Otomatis untuk Proses Sterilisasi Susu dengan Metode Konvensional Menggunakan Kontrol PID Berbasis Arduino Mega 2560 dengan Telemetri.*

3.7.1 Kriteria Pengujian Perangkat Keras Sistem Kendali

3.7.1.1 Pengujian Rangkaian LCD 16x4

Pengujian rangkaian LCD dilakukan untuk mengetahui apakah rangkaian LCD 16x4 dapat bekerja dengan baik?. Pengujian ini dilakukan dengan cara menghubungkan rangkaian LCD pada Arduino Mega 2560 yang telah diprogram dengan menggunakan program LCD untuk Arduino.

Tabel 3.1 Kriteria Pengujian Rangkaian LCD 16x4

Kriteria yang diuji	Kriteria Pengujian	Hasil Pengujian
Lampu Latar	Menyala	
Kontras	Sangat jelas	
Menampilkan Karakter	Berhasil	

3.7.1.2 Pengujian Rangkaian *Keypad*

Pengujian rangkaian *keypad* dilakukan untuk mengetahui apakah setiap tombol dapat berfungsi dengan baik?. Pengujian ini dilakukan dengan cara menghubungkan rangkaian *keypad* dengan Arduino Mega 2560 yang telah diprogram untuk mendeteksi sinyal *HIGH* (1) yang berasal dari rangkaian *keypad* ketika tombol ditekan, dan mendeteksi sinyal *LOW* (0) yang berasal dari rangkaian *keypad* ketika tombol tidak ditekan, pengujian ini dilakukan dengan menggunakan LCD 16x4.

Berikut adalah kriteria pengujian yang akan dilakukan pada rangkaian *keypad* dengan menggunakan LCD 16x4*:*

Tabel 3.2 Kriteria Pengujian Rangkaian *Keypad*

Kriteria yang diuji	Kriteria Pengujian	Hasil Pengujian
Tombol OK ditekan	*HIGH* (1)	
Tombol OK tidak ditekan	*LOW* (0)	
Tombol UP ditekan	*HIGH* (1)	
Tombol UP tidak ditekan	*LOW* (0)	
Tombol DOWN ditekan	*HIGH* (1)	
Tombol DOWN tidak ditekan	*LOW* (0)	
Tombol CANCEL ditekan	*HIGH* (1)	
Tombol CANCEL tidak ditekan	*LOW* (0)	

3.7.1.3 Pengujian Rangkaian *Low Pass Filter*

Pengujian rangkaian *low pass filter* untuk mengetahui apakah PWM keluaran dari Arduino Mega 2560 telah difilter dengan baik oleh rangkaian *low pass filter* atau tidak?. Pengujian ini dilakukan dengan cara menghubungkan *input* rangkaian *low pass filter* ke *output* PWM dari Arduino Mega 2560, kemudian tegangan dari *output* rangkaian *low pass filter* akan diukur dengan menggunakan AVO meter digital, besar PWM akan memepengaruhi besar tegangan *output* pada rangkaian *low pass filter*, berikut kriteria pengujiannya:

Tabel 3.3 Kritera Pengujian Rangkaian *Low Pass Filter*

Kriteria yang diuji	Kriteria Pengukuran	Hasil Pengukuran
PWM = 255	5 Volt	
PWM = 125	2.45 Volt	
PWM = 65	1.27 Volt	
PWM = 0	0 Volt	

3.7.1.4 Pengujian Rangkaian *Zero Crossing Detector*

Pengujian rangkaian *zero crossing detector* untuk mengetahui apakah rangkaian ini dapat mendeteksi tegangan nol pada sumber tegangan AC220V atau tidak. Pengujian ini dilakukan dengan cara menghubungkan *input* rangkaian *zero crossing detector* dengan sumber tegangan AC220V, kemudian *output* dari rangkaian *zero crossing detector* dihubungkan pada pin *interrupt* Arduino Nano. Pengujian ini dilakukan dengan menggunakan *serial monitor*. Berikut adalah kriteria pengujian yang akan dilakukan pada rangkaian *zero crossing detector* dengan menggunakan *serial monitor*:

Tabel 3.4 Kriteria Pengujian Rangkaian *Zero Crossing Detector*

Kriteria yang diuji	Kriteria Pengujian	Hasil Pengujian
Interupsi Arduino Nano	Aktif	

3.7.1.5 Pengujian Rangkaian *Driver Heater*

Pengujian rangkaian *driver heater* dilakukan untuk mengetahui apakah rangkaian *driver heater* dapat digunakan untuk mengatur besar daya yang akan dikirimkan pada *heater* atau tidak. Pengujian ini dilakukan dengan menghubungkan *input* dari rangkaian *driver heater* ke *output* PWM Arduino Nano, kemudian tegangan *output* dari rangakaian *driver heater* akan diukur dengan menggunakan AVO meter.

Berikut adalah kriteria pengujian yang akan dilakukan pada rangkaian *driver heater* dengan menggunakan AVO meter sebagai alat ukur tegangan *output* dari rangkaian:

Tabel 3.5 Kriteria Pengujian Rangkaian *Driver Heater*

Kriteria yang diuji	Kriteria Pengukuran	Hasil Pengukuran
PWM = 255	220 Volt	
PWM = 125	140,10 Volt	
PWM = 65	32,45 Volt	
PWM = 0	0 Volt	

3.7.1.6 Pengujian *Heater*

Pengujian ini dilakukan untuk mengetahui berapa waktu yang dibutuhkan oleh sistem dalam memanaskan suhu pada air susu dari suhu ruangan sebesar 30^0C hingga suhu maksimal sebesar 130^0C dan untuk mengetahui berapa waktu yang dibutuhkan oleh sistem dalam mendinginkan suhu pada air susu dari suhu maksimal sebesar 130^0C hingga suhu 100^0C. Berikut adalah kriteria pengujiannya:

Tabel 3.6 Kriteria Pengujian *Heater*

Kriteria yang diuji	Waktu yang dibutuhkan
Memanaskan suhu air dari 30^0C ke 130^0C	
Mendinginkan suhu air dari 130^0C ke 100^0C	

3.7.1.7 Pengujian Sensor Suhu MLX90614

Pengujian ini dilakukan untuk mengetahui berapa jarak ideal sensor MLX90614 terhadap objek yang diukur agar hasil pengukuran yang didapat menjadi akurat?. Pengujian dilakukan dengan cara membandingkan hasil pengukuran sensor MLX90614 terhadap objek yang akan diukur dengan hasil pengukuran dari termometer digital yang berstandar internasional.

Berikut adalah kriteria pengujian yang akan dilakukan terhadap jarak ideal yang digunakan antara sensor MLX90614 terhadap objek yang diukur lalu dibandingkan dengan pengukuran termometer digital:

Tabel 3.7 Kriteria Pengujian Sensor MLX90614

Kriteria yang diuji	Kriteria Pengukuran	Hasil Pengukuran
Jarak = 1 cm	Suhu = 30^0C	
Jarak = 2 cm	Suhu = 30^0C	
Jarak = 3 cm	Suhu = 30^0C	
Jarak = 4 cm	Suhu = 30^0C	
Jarak = 5 cm	Suhu = 30^0C	

3.7.2 Kriteria Pengujian Perangkat Lunak Sistem Kendali

Berikut adalah kriteria pengujian perangkat lunak dari sistem kendali secara keseluruhan:

Tabel 3.8 Pengujian Perangkat Lunak pada Arduino

Kriteria yang diuji	Kriteria Pengujian	Hasil Pengujian
Tombol OK	Berfungsi	
Tombol UP	Berfungsi	
Tombol DOWN	Berfungsi	
Tombol CANCEL	Berfungsi	
Tampilan pengaturan parameter pada LCD	Berhasil	
Tampilan data pembacaan suhu pada LCD	Berhasil	
Pengaturan daya pada *heater* secara otomatis	Berhasil	
Pengiriman data suhu melalui *Bluetooth*	Berhasil	
Data suhu terbaca pada *smartphone Android*	Berhasil	
Tombol *STOP* pada *Smartphone* Android	Berfungsi	

Tabel 3. 9 Hasil Pengujian Perangkat Lunak pada *Smartphone* Android

Kriteria yang diuji	Kriteria Pengujian	Hasil Pengujian
Pembacaan data suhu	Berhasil	
Tombol START	Berfungsi	
Tombol STOP	Berfungsi	
Tombol OK	Berfungsi	

3.7.3 Kriteria PengujianTelemetri (*Bluetooth*)

Pengujian ini dilakukan untuk mengetahui berapa jarak maksimal yang bisa digunakan agar *smartphone* Android dapat membaca data suhu yang dikirim melalui koneksi *Bluetooth*?, berikut adalah kriteria pengujiannya:

Tabel 3.10 Kriteria Pengujian Telemetri

Kriteria yang diuji	Kriteria Pengujian	Hasil Pengujian
Jarak = 0,5 meter	Data suhu terbaca	
Jarak = 1 meter	Data suhu terbaca	
Jarak = 5 meter	Data suhu terbaca	
Jarak = 10 meter	Data suhu terbaca	
Jarak = 11 meter	Data suhu terbaca	

3.7.4 Kriteria Pengujian Kestabilan Suhu

Pengujian ini dilakukan untuk mengetahui apakah sistem kontrol suhu yang telah dibuat dapat menjaga kestabilan suhu sebesar 120^0C selama selang waktu 15 menit pada proses sterilisasi susu?. Pengujian ini dilakukan dengan melakukan tuning terhadap parameter-parameter PID dengan 5 kali percobaan sebagai berikut:

1. Percobaan Ke-1 dengan Nilai KP = 10, KI = 10, KD = 30

Tabel 3.11 Kriteria Pengujian Kestabilan Suhu Percobaan Ke-1

Kriteria yang diuji	Kriteria Pengujian	Hasil Pengujian
Suhu pada menit ke-1	120^0C	
Suhu pada menit ke-2	120^0C	
Suhu pada menit ke-3	120^0C	
Suhu pada menit ke-4	120^0C	
Suhu pada menit ke-5	120^0C	
Suhu pada menit ke-6	120^0C	
Suhu pada menit ke-7	120^0C	
Suhu pada menit ke-8	120^0C	
Suhu pada menit ke-9	120^0C	
Suhu pada menit ke-10	120^0C	
Suhu pada menit ke-11	120^0C	
Suhu pada menit ke-12	120^0C	
Suhu pada menit ke-13	120^0C	
Suhu pada menit ke-14	120^0C	
Suhu pada menit ke-15	120^0C	
Suhu rata-rata		

2. Percobaan Ke-2 dengan Nilai KP = 7, KI = 10, KD = 30

Tabel 3.12 Kriteria Pengujian Kestabilan Suhu Percobaan Ke-2

Kriteria yang diuji	Kriteria Pengujian	Hasil Pengujian
Suhu pada menit ke-1	120^0C	
Suhu pada menit ke-2	120^0C	
Suhu pada menit ke-3	120^0C	
Suhu pada menit ke-4	120^0C	
Suhu pada menit ke-5	120^0C	
Suhu pada menit ke-6	120^0C	
Suhu pada menit ke-7	120^0C	
Suhu pada menit ke-8	120^0C	
Suhu pada menit ke-9	120^0C	
Suhu pada menit ke-10	120^0C	
Suhu pada menit ke-11	120^0C	
Suhu pada menit ke-12	120^0C	
Suhu pada menit ke-13	120^0C	
Suhu pada menit ke-14	120^0C	
Suhu pada menit ke-15	120^0C	
Suhu rata-rata		

3. Percobaan Ke-3 dengan Nilai KP = 7, KI = 7, KD = 30

Tabel 3.13 Kriteria Pengujian Kestabilan Suhu Percobaan Ke-3

Kriteria yang diuji	Kriteria Pengujian	Hasil Pengujian
Suhu pada menit ke-1	120^0C	
Suhu pada menit ke-2	120^0C	
Suhu pada menit ke-3	120^0C	
Suhu pada menit ke-4	120^0C	
Suhu pada menit ke-5	120^0C	
Suhu pada menit ke-6	120^0C	
Suhu pada menit ke-7	120^0C	
Suhu pada menit ke-8	120^0C	
Suhu pada menit ke-9	120^0C	
Suhu pada menit ke-10	120^0C	
Suhu pada menit ke-11	120^0C	
Suhu pada menit ke-12	120^0C	
Suhu pada menit ke-13	120^0C	
Suhu pada menit ke-14	120^0C	
Suhu pada menit ke-15	120^0C	
Suhu rata-rata		

4. Percobaan Ke-4 dengan Nilai KP = 7, KI = 7, KD = 40

Tabel 3 14 Kriteria Pengujian Kestabilan Suhu Percobaan Ke-4

Kriteria yang diuji	Kriteria Pengujian	Hasil Pengujian
Suhu pada menit ke-1	120^0C	
Suhu pada menit ke-2	120^0C	
Suhu pada menit ke-3	120^0C	
Suhu pada menit ke-4	120^0C	
Suhu pada menit ke-5	120^0C	
Suhu pada menit ke-6	120^0C	
Suhu pada menit ke-7	120^0C	
Suhu pada menit ke-8	120^0C	
Suhu pada menit ke-9	120^0C	
Suhu pada menit ke-10	120^0C	
Suhu pada menit ke-11	120^0C	
Suhu pada menit ke-12	120^0C	
Suhu pada menit ke-13	120^0C	
Suhu pada menit ke-14	120^0C	
Suhu pada menit ke-15	120^0C	
Suhu rata-rata		

5. Percobaan Ke-5 dengan Nilai KP = 7, KI = 7, KD = 50

Tabel 3.15 Kriteria Pengujian Kestabilan Suhu Percobaan Ke-5

Kriteria yang diuji	Kriteria Pengujian	Hasil Pengujian
Suhu pada menit ke-1	120^0C	
Suhu pada menit ke-2	120^0C	
Suhu pada menit ke-3	120^0C	
Suhu pada menit ke-4	120^0C	
Suhu pada menit ke-5	120^0C	
Suhu pada menit ke-6	120^0C	
Suhu pada menit ke-7	120^0C	
Suhu pada menit ke-8	120^0C	
Suhu pada menit ke-9	120^0C	
Suhu pada menit ke-10	120^0C	
Suhu pada menit ke-11	120^0C	
Suhu pada menit ke-12	120^0C	
Suhu pada menit ke-13	120^0C	
Suhu pada menit ke-14	120^0C	
Suhu pada menit ke-15	120^0C	
Suhu rata-rata		

BAB IV

HASIL PENELITIAN DAN PEMBAHASAN

4.1 Hasil Penelitian

Hasil pengujian yang dilakukan terhadap indikator-indikator penelitian dari "Sistem Kontrol Suhu Otomatis untuk Proses Sterilisasi Susu dengan Metode Konvensional dengan Telemetri" dapat dikategorikan sebagai berikut:

1. Hasil pengujian perangkat keras sistem kendali.
2. Hasil pengujian perangkat lunak sistem kendali.
3. Hasil pengujian telemetri.
4. Hasil pengujian sistem kontrol suhu otomatis dalam menjaga kestabilan suhu selama proses sterilisasi susu dengan 5 kali percobaan

4.1.1 Hasil Pengujian Perangkat Keras (*Hardware*)

4.1.1.1 Pengujian Rangkaian LCD 16x4

Pengujian pada rangkaian LCD 16x4 ini dilakukan untuk mengetahui apakah lampu latar pada LCD berfungsi, kontras layar pada LCD sudah sangat jelas, dan LCD dapat menampilkan karakter dengan benar sesuai dengan program yang telah dibuat, berikut adalah tabel hasil pengujian rangkaian LCD 16x4:

Tabel 4.1 Hasil Pengujian Rangkaian LCD 16x4

Kriteria yang diuji	Kriteria Pengujian	Hasil Pengujian
Lampu Latar	Menyala	Menyala
Kontras	Sangat jelas	Sangat Jelas
Menampilkan Karakter	Berhasil	Berhasil

Gambar 4.1 Hasil Pengujiaan LCD 16x4

4.1.1.2 Pengujian Rangkaian *Keypad*

Pengujian rangkaian *keypad* ini bertujuan untuk mengetahui apakah rangkaian *keypad* sudah berfungsi dengan benar atau belum sesuai dengan program yang telah dibuat, berikut adalah hasil pengujian rangkaian *keypad* yang telah dilakukan:

Tabel 4.2 Hasil Pengujian Rangkaian *Keypad*

Kriteria yang diuji	Kriteria Pengujian	Hasil Pengujian
Tombol OK ditekan	*HIGH* (1)	*HIGH* (1)
Tombol OK tidak ditekan	*LOW* (0)	*LOW* (0)
Tombol UP ditekan	*HIGH* (1)	*HIGH* (1)
Tombol UP tidak ditekan	*LOW* (0)	*LOW* (0)
Tombol DOWN ditekan	*HIGH* (1)	*HIGH* (1)
Tombol DOWN tidak ditekan	*LOW* (0)	*LOW* (0)
Tombol CANCEL ditekan	*HIGH* (1)	*HIGH* (1)
Tombol CANCEL tidak ditekan	*LOW* (0)	*LOW* (0)

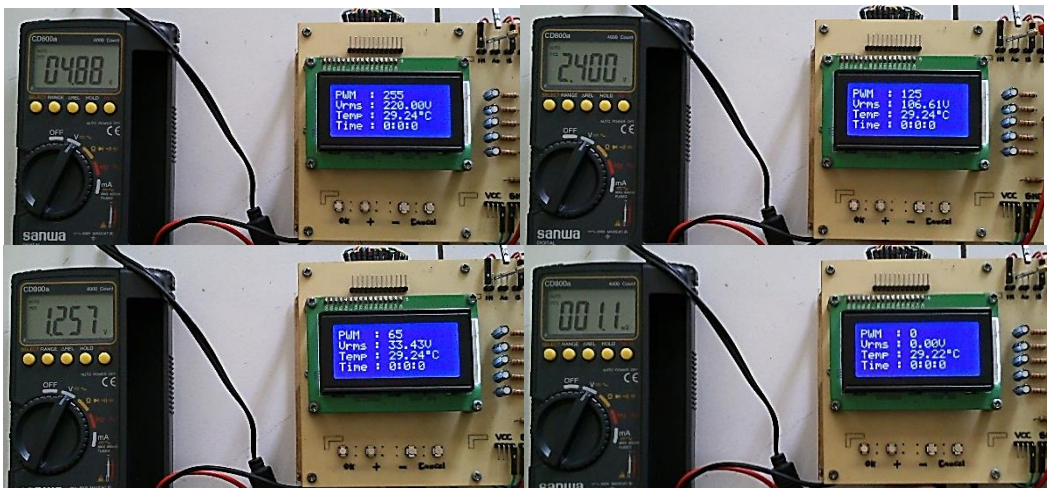
Gambar 4.2 Hasil Pengujian Rangkaian *Keypad*

4.1.1.3 Pengujian Rangkaian *Low Pass Filter*

Pengujian rangkaian *low pass filter* untuk mengetahui apakah PWM keluaran dari Arduino Mega 2560 telah difilter dengan baik oleh rangkaian *low pass filter* atau tidak?. Pengujian ini dilakukan dengan cara menghubungkan *input* rangkaian *low pass filter* ke *output* PWM dari Arduino Mega 2560, kemudian tegangan dari *output* rangkaian *low pass filter* akan diukur dengan menggunakan AVO meter digital, besar PWM akan memepengaruhi besar tegangan *output* pada rangkaian *low pass filter* yang akan diukur, berikut kriteria pengujiannya:

Tabel 4.3 Hasil Pengujian Rangkaian *Low Pass Filter*

Kriteria yang diuji	Kriteria Pengujian	Hasil Pengujian
PWM = 255	5 Volt	4.88 Volt
PWM = 125	2.45 Volt	2.40 Volt
PWM = 65	1.27 Volt	1.26 Volt
PWM = 0	0 Volt	0 Volt



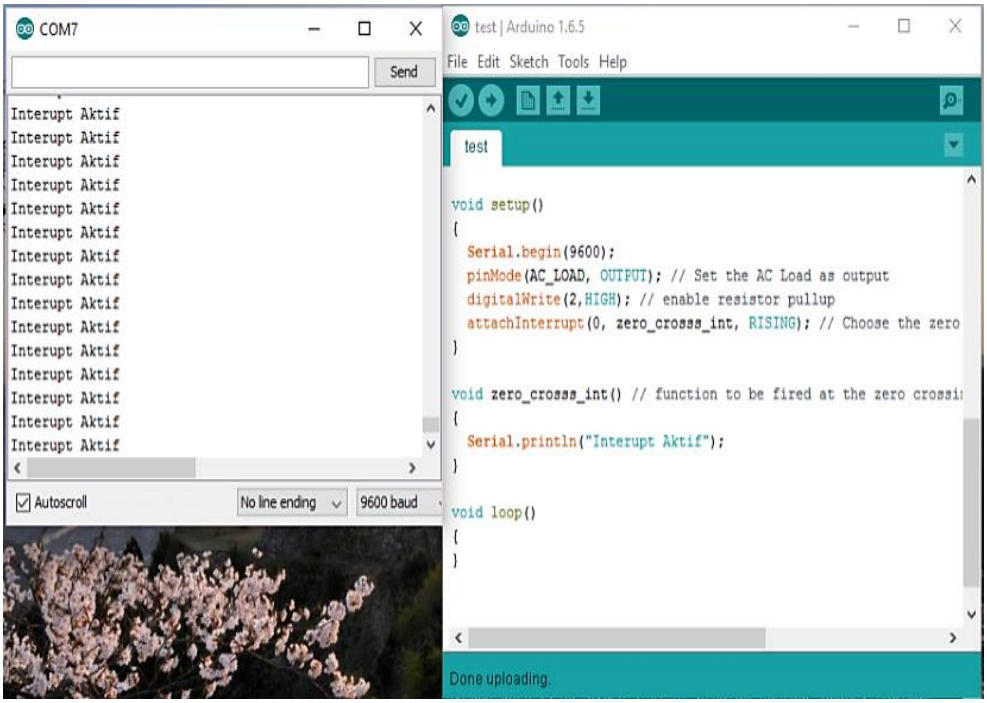
Gambar 4.3 Hasil Pengujian Rangkaian *Low Pass Filter*

4.1.1.4 Pengujian Rangkaian *Zero Crossing Detector*

Pengujian rangkaian *zero crossing detector* ini dilkukan untuk mendapatkan pendeteksian tegangan nol pada sumber AC220V, jika pendeteksian tengangan nol pada sumber AC220V berhasil maka program interupsi pada Arduino Nano akan berjalan, berikut adalah hasil pengujian rangkaian *zero crossing detector* yang telah dilakukan:

Tabel 4.4 Hasil Pengujian Rangkaian *Zero Crossing Detector*

Kriteria yang diuji	Kriteria Pengujian	Hasil Pengujian
Interupsi Arduino Nano	Aktif	Aktif



Gambar 4.4 Hasil Pengujian Rangkaian *Zero Crossing Detector*

4.1.1.5 Pengujian Rangkaian *Driver Heater*

Pengujian rangkaian *driver heater* ini dilakukan untuk mendapatkan besar tegangan *output* dari rangkaian *driver heater* sesuai dengan nilai PWM (*pulse width modulation*) yang diberikan pada *input* rangkaian, besar tegangan akan berubah-ubah ketika nilai PWM (*pulse width modulation*) juga ikut diubah, berikut adalah hasil pengujian rangkaian *driver heater* yang telah dilakukan:

Tabel 4.5 Hasil Pengujian Rangkaian *Driver Heater*

Kriteria yang diuji	Kriteria Pengukuran	Hasil Pengukuran
PWM = 255	220.00 Volt	212.00Volt
PWM = 125	106,6 Volt	106.0 Volt
PWM = 65	33,42 Volt	32.0 Volt
PWM = 0	0 Volt	0 Volt

Gambar 4.5 Hasil Pengujian Rangkaian *Driver Heater*

4.1.1.6 Pengujian *Heater*

Pengujian ini dilakukan untuk mengetahui berapa waktu yang dibutuhkan oleh sistem dalam memanaskan suhu pada air susu dari suhu ruangan sebesar 30^0C hingga suhu maksimal sebesar 130^0C dan untuk mengetahui berapa waktu yang dibutuhkan oleh sistem dalam mendinginkan suhu pada air susu dari suhu maksimal sebesar 130^0C hingga suhu 100^0C. Berikut adalah hasil pengujiannya:

Tabel 4.6 Hasil Pengujian *Heater*

Kriteria yang diuji	Waktu yang dibutuhkan
Memanaskan suhu air dari 30^0C ke 130^0C	18 Menit
Mendinginkan suhu air dari 130^0C ke 100^0C	38 Menit

4.1.1.7 Pengujian Sensor Suhu MLX90614

Pengujian sensor suhu MLX90614 ini dilakukan untuk mendapatkan jarak ideal yang digunakan agar sensor dapat membaca suhu secara akurat, berikut adalah hasil pengujian sensor suhu MLX90614 yang telah dilakukan:

Tabel 4.7 Hasil Pengujian Sensor Suhu MLX90614

Kriteria yang diuji	Kriteria Pengukuran	Hasil Pengukuran
Jarak = 1 cm	Suhu = 30^0C	Suhu = 30.20^0C
Jarak = 2 cm	Suhu = 30^0C	Suhu = 30.14^0C
Jarak = 3 cm	Suhu = 30^0C	Suhu = 30.12^0C
Jarak = 4 cm	Suhu = 30^0C	Suhu = 30.10^0C
Jarak = 5 cm	Suhu = 30^0C	Suhu = 30.06^0C

4.1.2 Hasil Pengujian Perangkat Lunak Sistem Kendali

Pengujian perangkat lunak sistem kendali ini dilakukan untuk mengetahui apakah perangkat lunak yang telah dibuat dapat berjalan dengan baik, berikut adalah hasil pengujian yang telah dilakukan:

Tabel 4.8 Hasi Pengujian Perangkat Lunak pada Arduino

Kriteria yang diuji	Kriteria Pengukuran	Hasil Pengukuran
Tombol OK	Berfungsi	Berfungsi
Tombol UP	Berfungsi	Berfungsi
Tombol DOWN	Berfungsi	Berfungsi
Tombol CANCEL	Berfungsi	Berfungsi
Tampilan pengaturan parameter pada LCD	Berhasil	Berhasil
Tampilan data pembacaan suhu pada LCD	Berhasil	Berhasil
Pengaturan daya pada *heater* secara otomatis	Berhasil	Berhasil
Pengiriman data suhu melalui *Bluetooth*	Berhasil	Berhasil

Gambar 4.6 Hasil Pengujian Perangkat Lunak Sistem Kendali

Tabel 4.9 Hasil Pengujian Perangkat Lunak pada *Smartphone* Android

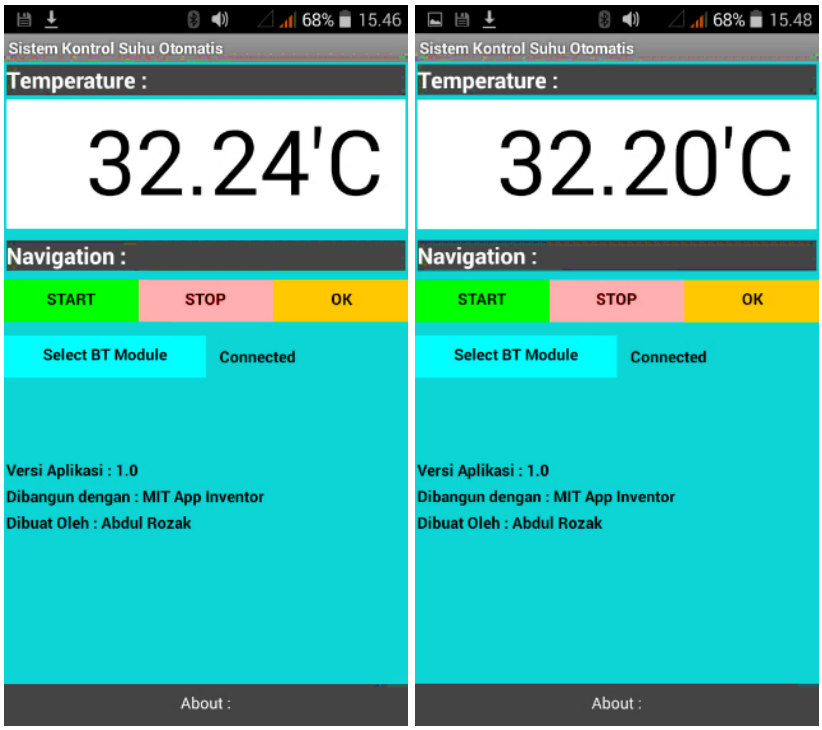
Kriteria yang diuji	Kriteria Pengujian	Hasil Pengujian
Pembacaan data suhu	Berhasil	Berhasil
Tombol START	Berfungsi	Berfungsi
Tombol STOP	Berfungsi	Berfungsi
Tombol OK	Berfungsi	Berfungsi

4.1.3 Pengujian Telemetri (*Bluetooth*)

Pengujian telemetri dilakukan untuk mendapatkan maksimal radius jarak yang dapat digunakan agar *smartphone* Android dengan Arduino dapat berkomunikasi dengan baik melalui koneksi *Bluetooth*, berikut adalah hasil pengujian telemetri:

Tabel 4.10 Hasil Pengujian Telemetri

Kriteria yang diuji	Kriteria Pengujian	Hasil Pengujian
Jarak = 0,5 meter	Data suhu terbaca	Data suhu terbaca
Jarak = 1 meter	Data suhu terbaca	Data suhu terbaca
Jarak = 5 meter	Data suhu terbaca	Data suhu terbaca
Jarak = 10 meter	Data suhu terbaca	Data suhu terbaca
Jarak = 11 meter	Data suhu terbaca	Data suhu tidak terbaca



Gambar 4.7 Hasil Pengujian Telemetri

4.1.4 Hasil Pengujian Sistem Kontrol Suhu Otomatis dalam Menjaga Kestabilan Suhu selama Proses Sterilisasi Susu

Hasil pengujian ini dilakukan untuk mendapatkan sistem kontrol suhu yang stabil, pengujian ini dilakukan dengan melakukan tuning pada parameter-parameter PID sebanyak 5 kali percobaan hingga didapatkan sistem kontrol suhu yang paling stabil, berikut adalah hasil pengujiannya:

1. Percobaan Ke-1 dengan Nilai KP = 10, KI = 10, KD = 30

Tabel 4.11 Hasil Pengujian Kestabilan Suhu Percobaan Ke-1

Kriteria yang diuji	Kriteria Pengujian	Hasil Pengujian
Suhu pada menit ke-1	120^0C	$123,00^0$C
Suhu pada menit ke-2	120^0C	$125,50^0$C
Suhu pada menit ke-3	120^0C	$125,40^0$C
Suhu pada menit ke-4	120^0C	$125,30^0$C
Suhu pada menit ke-5	120^0C	$125,30^0$C
Suhu pada menit ke-6	120^0C	$124,00^0$C
Suhu pada menit ke-7	120^0C	$123,50^0$C
Suhu pada menit ke-8	120^0C	$120,36^0$C
Suhu pada menit ke-9	120^0C	$118,40^0$C
Suhu pada menit ke-10	120^0C	$120,70^0$C
Suhu pada menit ke-11	120^0C	$122,00^0$C
Suhu pada menit ke-12	120^0C	$121,70^0$C
Suhu pada menit ke-13	120^0C	$118,70^0$C
Suhu pada menit ke-14	120^0C	$120,40^0$C
Suhu pada menit ke-15	120^0C	$120,58^0$C
Suhu rata-rata		$122,32^0$C

2. Percobaan Ke-2 dengan Nilai KP = 7, KI = 10, KD = 30

Tabel 4.12 Hasil Pengujian Kestabilan Suhu Percobaan Ke-2

Kriteria yang diuji	Kriteria Pengujian	Hasil Pengujian
Suhu pada menit ke-1	120^0C	$120,70^0$C
Suhu pada menit ke-2	120^0C	$126,00^0$C
Suhu pada menit ke-3	120^0C	$125,90^0$C
Suhu pada menit ke-4	120^0C	$125,50^0$C
Suhu pada menit ke-5	120^0C	$124,20^0$C
Suhu pada menit ke-6	120^0C	$122,80^0$C
Suhu pada menit ke-7	120^0C	$121,80^0$C
Suhu pada menit ke-8	120^0C	$119,00^0$C
Suhu pada menit ke-9	120^0C	$120,10^0$C
Suhu pada menit ke-10	120^0C	$119,60^0$C
Suhu pada menit ke-11	120^0C	$120,30^0$C
Suhu pada menit ke-12	120^0C	$119,00^0$C
Suhu pada menit ke-13	120^0C	$121,70^0$C
Suhu pada menit ke-14	120^0C	$121,20^0$C
Suhu pada menit ke-15	120^0C	$119,90^0$C
Suhu rata-rata		121.35^0C

3. Percobaan Ke-3 dengan Nilai KP = 7, KI = 7, KD = 30

Tabel 4.13 Hasil Pengujian Kestabilan Suhu Percobaan Ke-3

Kriteria yang diuji	Kriteria Pengujian	Hasil Pengujian
Suhu pada menit ke-1	120^0C	$120,18^0$C
Suhu pada menit ke-2	120^0C	$120,58^0$C
Suhu pada menit ke-3	120^0C	119.50^0C
Suhu pada menit ke-4	120^0C	120.02^0C
Suhu pada menit ke-5	120^0C	$119,62^0$C
Suhu pada menit ke-6	120^0C	122.20^0C
Suhu pada menit ke-7	120^0C	$121,00^0$C
Suhu pada menit ke-8	120^0C	$119,50^0$C
Suhu pada menit ke-9	120^0C	$118,20^0$C
Suhu pada menit ke-10	120^0C	$121,00^0$C
Suhu pada menit ke-11	120^0C	$123,00^0$C
Suhu pada menit ke-12	120^0C	$120,70^0$C
Suhu pada menit ke-13	120^0C	$120,84^0$C
Suhu pada menit ke-14	120^0C	$119,50^0$C
Suhu pada menit ke-15	120^0C	$120,38^0$C
Suhu rata-rata		120.40^0C

4. Percobaan Ke-4 dengan Nilai KP = 7, KI = 7, KD = 40

Tabel 4.14 Hasil Pengujian Kestabilan Suhu Percobaan Ke-4

Kriteria yang diuji	Kriteria Pengujian	Hasil Pengujian
Suhu pada menit ke-1	120^0C	$120,80^0$C
Suhu pada menit ke-2	120^0C	$120,50^0$C
Suhu pada menit ke-3	120^0C	$121,50^0$C
Suhu pada menit ke-4	120^0C	$118,60^0$C
Suhu pada menit ke-5	120^0C	$120,20^0$C
Suhu pada menit ke-6	120^0C	$119,20^0$C
Suhu pada menit ke-7	120^0C	$119,80^0$C
Suhu pada menit ke-8	120^0C	$119,90^0$C
Suhu pada menit ke-9	120^0C	$119,34^0$C
Suhu pada menit ke-10	120^0C	$120,66^0$C
Suhu pada menit ke-11	120^0C	$120,80^0$C
Suhu pada menit ke-12	120^0C	$120,40^0$C
Suhu pada menit ke-13	120^0C	$118,20^0$C
Suhu pada menit ke-14	120^0C	$121,40^0$C
Suhu pada menit ke-15	120^0C	$121,80^0$C
Suhu rata-rata		$120,31^0$C

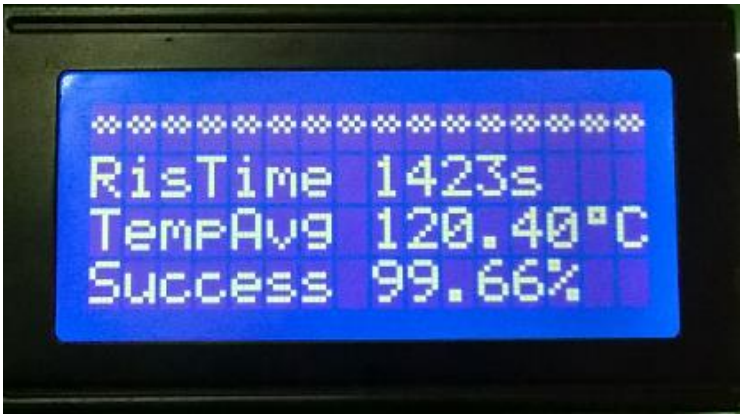
5. Percobaan Ke-5 dengan Nilai KP = 7, KI = 7, KD = 50

Tabel 4.15 Hasil Pengujian Kestabilan Suhu Percobaan Ke-5

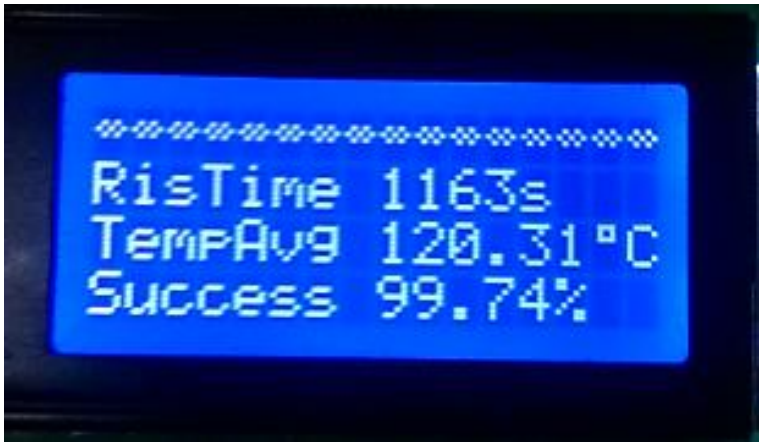
Kriteria yang diuji	Kriteria Pengukuran	Hasil Pengukuran
Suhu pada menit ke-1	120^0C	$120,40^0$C
Suhu pada menit ke-2	120^0C	$121,40^0$C
Suhu pada menit ke-3	120^0C	$121,50^0$C
Suhu pada menit ke-4	120^0C	$120,70^0$C
Suhu pada menit ke-5	120^0C	$119,10^0$C
Suhu pada menit ke-6	120^0C	$119,60^0$C
Suhu pada menit ke-7	120^0C	$118,66^0$C
Suhu pada menit ke-8	120^0C	$119,00^0$C
Suhu pada menit ke-9	120^0C	$122,00^0$C
Suhu pada menit ke-10	120^0C	$122,40^0$C
Suhu pada menit ke-11	120^0C	$122,30^0$C
Suhu pada menit ke-12	120^0C	$119,00^0$C
Suhu pada menit ke-13	120^0C	$120,50^0$C
Suhu pada menit ke-14	120^0C	$118,90^0$C
Suhu pada menit ke-15	120^0C	$119,20^0$C
Suhu rata-rata		120.46^0C

Gambar 4.8 Hasil Percobaan Ke-1

Gambar 4.9 Hasil Percobaan Ke-2

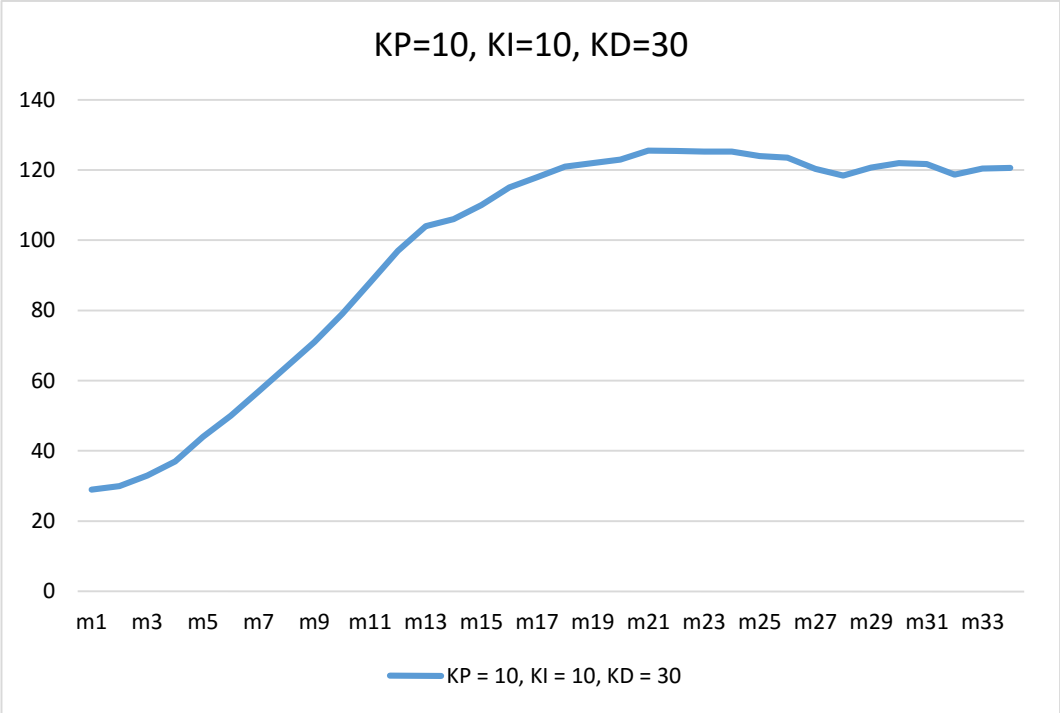


Gambar 4.10 Hasil Percobaan Ke-3

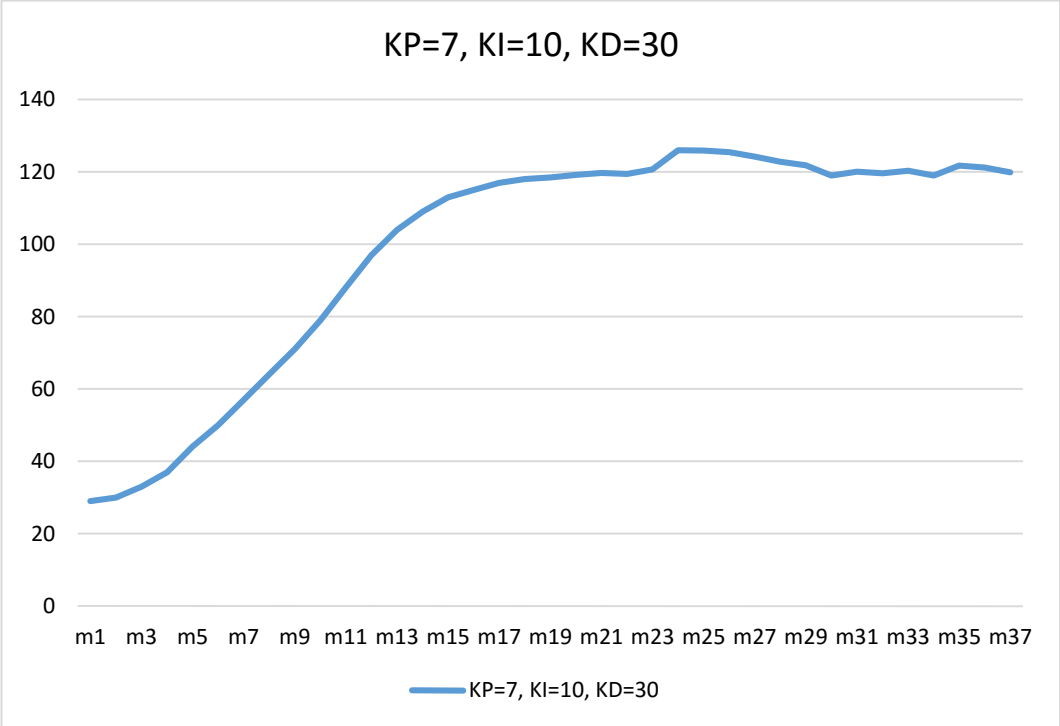


Gambar 4.11 Hasil Percobaan Ke-4

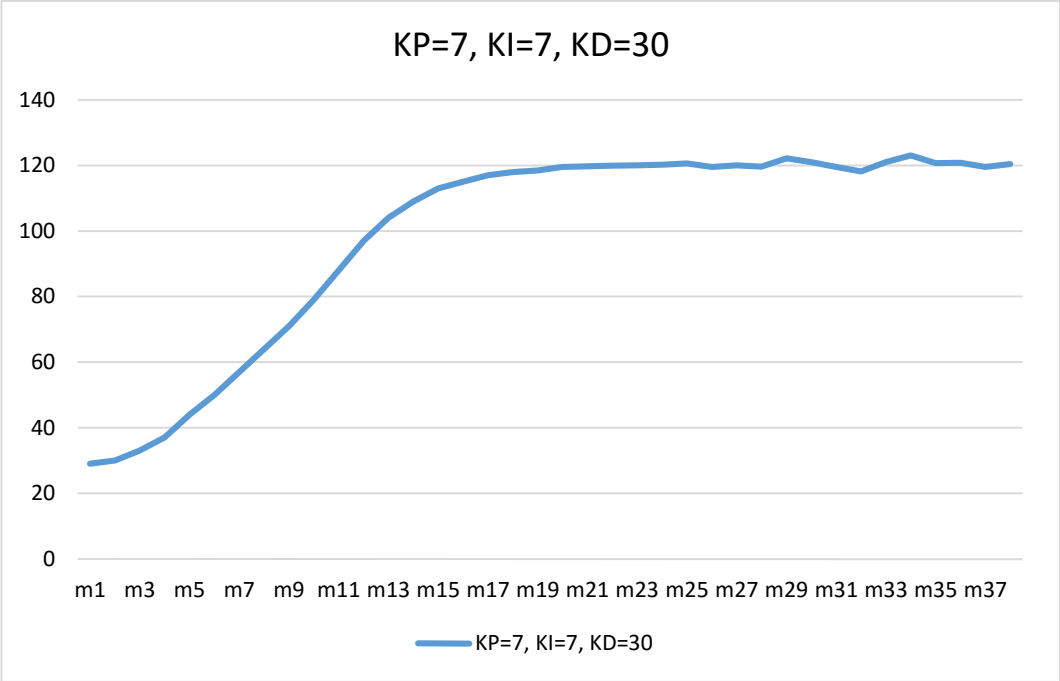
Gambar 4.12 Hasil Percobaan Ke-5



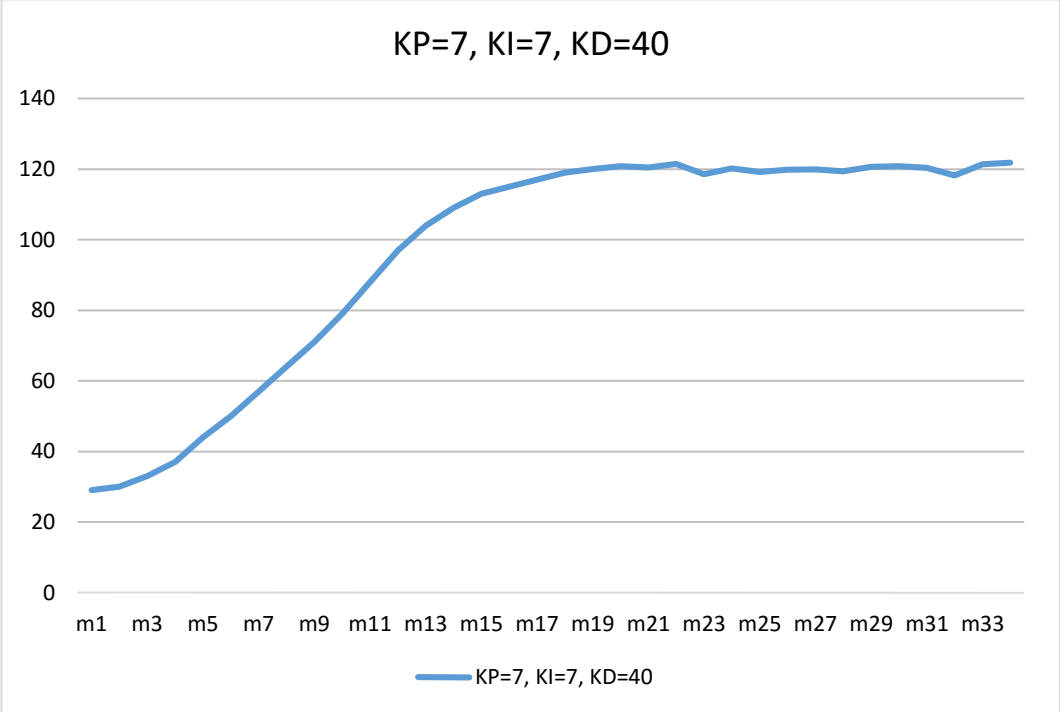
Gambar 4.13 Grafik Kestabilan Suhu Percobaan ke-1



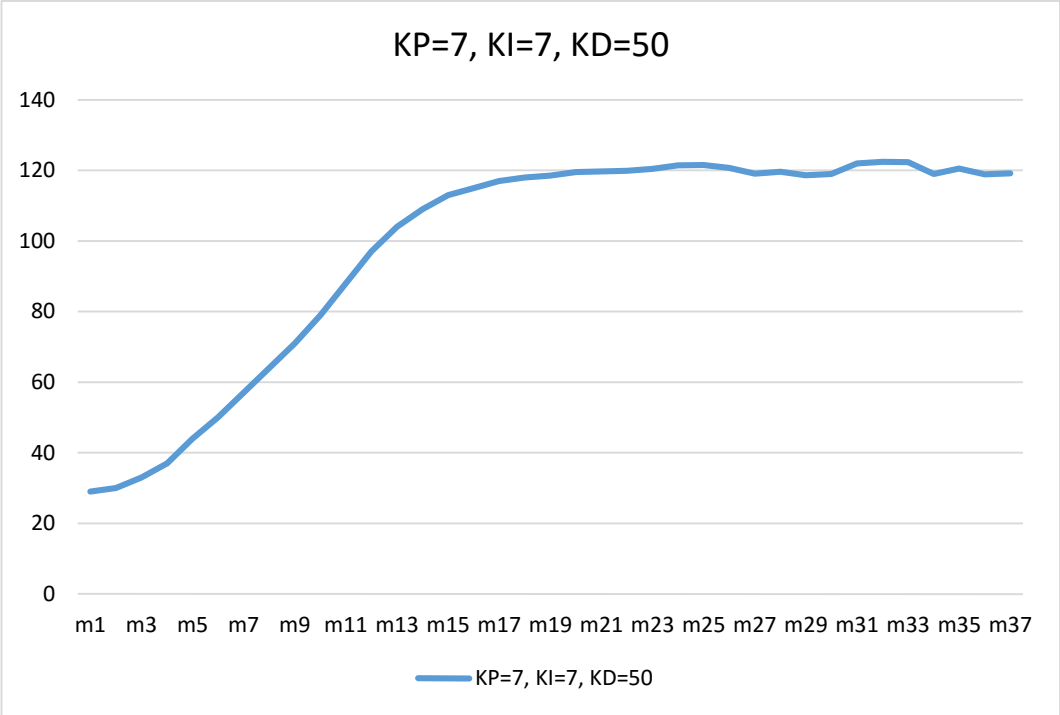
Gambar 4.14 Grafik Kestabilan Suhu Percobaan Ke-2



Gambar 4.15 Grafik Kestabilan Suhu Percobaan Ke-3



Gambar 4.16 Grafik Kestabilan Suhu Percobaan Ke-4



Gambar 4.17 Grakfik Kestabilan Suhu Percobaan Ke-5

4.2 Pembahasan

Berdasarkan dengan pengujian yang telah dilakukan, *Sistem Kendali Suhu Otomatis untuk Proses Sterilisasi Susu dengan Metode Konvensional Mengunakan Kontrol PID Berbasis Arduino Mega 2560 dengan Telemetri*, hasil percobaan ke-4 merupakan yang paling stabil, yaitu dengan suhu rata-rata sebesar $120{,}31^0$C atau dengan persentasi *error*:

$$Persentasi\ error = \frac{120{,}31^0C - 120^0C}{120^0C} = 0{,}26\%$$

Terdapatnya sedikit *error* pada sistem kendali dalam menjaga kestabilan suhu pada saat proses sterilisasi susu tersebut terjadi karena tidak tepatnya pemilihan parameter-parameter PID (*Proportional Integral Derivative*) untuk sistem kendali karena pemilihannya dilakukan dengan metode coba-coba (*trial &*

error) dan terdapatnya gangguan dari luar sistem berupa suhu ruangan yang berubah-ubah dan sumber listrik PLN yang tidak stabil sehingga menyebabkan sistem kendali suhu menjadi tidak stabil.

Dari hasil percobaan kestabilan suhu yang telah dilakukan sebanyak 5 kali, dapat dilihat pada percobaaan ke-4 merupakan yang paling stabil dengan suhu rata-rata sebesar 120,31^{0}C atau memiliki persentasi error sebesar 0,26%. Nilai dari parmeter-parameter PID yang digunakan pada percobaan ke-4 adalah KP = 7, KI = 7, dan KD = 40.

Berikut adalah perhitungan dari kontrol PID pada percobaan ke-4:

$Suhu\ yang\ dibutuhkan = 120^0C$

$KP = 7, KI = 7, KD = 40$

$Waktu\ cuplik\ sensor, s = 10\ ms$

***Misalkan suhu pada air susu mencapai* $= \mathbf{100^0C}$ *maka*:**

$Error = suhu\ yang\ dibutuhkan - suhu\ pada\ air\ susu$

$Error = 120^0C - 100^0C = 20$

$Aksi\ kontrol\ Proportional = KP * error$

$Aksi\ kontrol\ Proportional = 7 * 20$

$Aksi\ kontrol\ Proportional = 140$

$Aksi\ kontrol\ Integral = \frac{KI}{s} * (error\ sekarang + error\ yang\ lalu)$

$Misalkan\ error\ yang\ lalu = 20.1\ , maka$:

$Aksi\ kontrol\ Integral = \frac{7}{10} * (20 + 20{,}1)$

$Aksi\ kontrol\ Integral = 28{,}07$

$Aksi\ kontrol\ Derivative = KD * s(error\ sekarang - error\ yaang\ lalu)$

$Aksi\ kontrol\ Derivative = 40 * 10 * (20 - 20{,}1)$

$Aksi\ kontrol\ Derivative = -40$

$Kontrol\ PID = P + I + D$

$Kontrol\ PID = 140 + 28{,}07 + (-40)$

$Kontrol\ PID = 128.07$

$Misalkan\ suhu\ pada\ air\ susu\ mencapai = 119^0C, maka$:

$Error = suhu\ yang\ dibutuhkan - suhu\ pada\ air\ susu$

$Error = 120^0C - 119^0C = 1$

$Aksi\ kontrol\ Proportional = KP * error$

$Aksi\ kontrol\ Proportional = 7 * 1$

$Aksi\ kontrol\ Proportional = 7$

$Aksi\ kontrol\ Integral = \frac{KI}{s} * (error\ sekarang + error\ yang\ lalu)$

$Misalkan\ error\ yang\ lalu = 1.1\ , maka$:

$Aksi\ kontrol\ Integral = \frac{7}{10} * (1 + 1{,}1)$

$Aksi\ kontrol\ Integral = 1.47$

$Aksi\ kontrol\ Derivative = KD * s(error\ sekarang - error\ yaang\ lalu)$

$Aksi\ kontrol\ Derivative = 40 * 10 * (1 - 1{,}1)$

$Aksi\ kontrol\ Derivative = -40$

$Kontrol\ PID = P + I + D$

$Kontrol\ PID = 7 + 1{,}47 + (-40)$

$Kontrol\ PID = -31.53$

$Misalkan\ suhu\ pada\ air\ susu\ mencapai = 120^0C, maka$:

$Error = suhu\ yang\ dibutuhkan - suhu\ pada\ air\ susu$

$Error = 120^0C - 120^0C = 0$

$Aksi\ kontrol\ Proportional = KP * error$

$Aksi\ kontrol\ Proportional = 7 * 0$

$$Aksi\ kontrol\ Proportional = 0$$

$$Aksi\ kontrol\ Integral = \frac{KI}{s} * (error\ sekarang + error\ yang\ lalu)$$

$$Misalkan\ error\ yang\ lalu = 0\ , maka:$$

$$Aksi\ kontrol\ Integral = \frac{7}{10} * (0 + 0)$$

$$Aksi\ kontrol\ Integral = 0$$

$$Aksi\ kontrol\ Derivative = KD * s(error\ sekarang - error\ yaang\ lalu)$$

$$Aksi\ kontrol\ Derivative = 40 * 10 * (0 - 0)$$

$$Aksi\ kontrol\ Derivative = 0$$

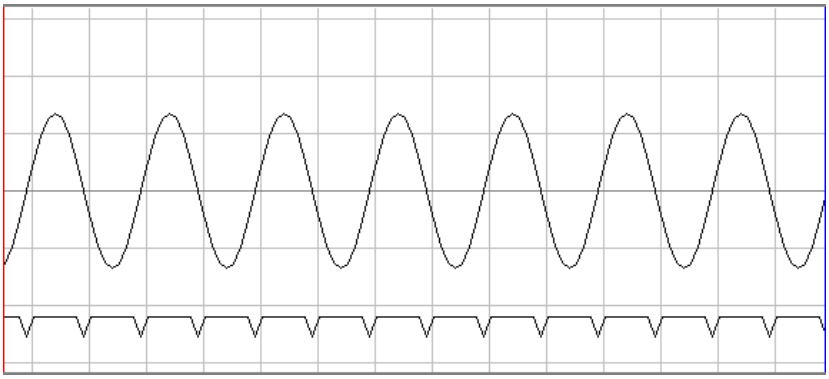
$$Kontrol\ PID = P + I + D$$

$$Kontrol\ PID = 0 + 0 + 0$$

$$Kontrol\ PID = 0$$

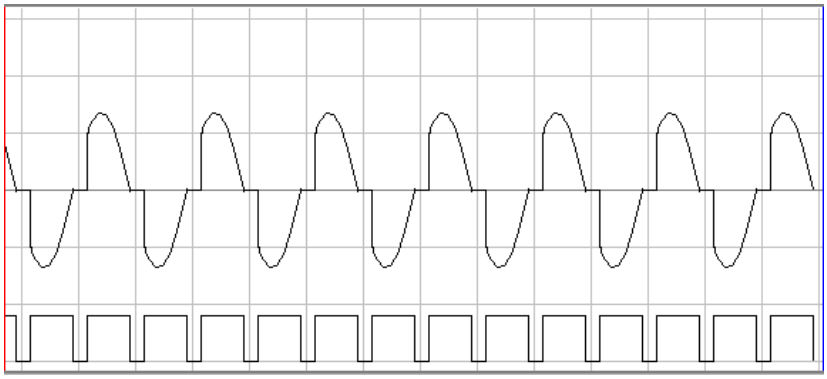
Berikut ini adalah grafik dari kontrol tegangan AC dengan menggunakan PWM (*Pulse Width Modulation*) berdasarkan aksi dari kontrol PID:

1. Ketika suhu mencapai 30^0C – 107^0C, PWM yang dihasilkan sebesar 255



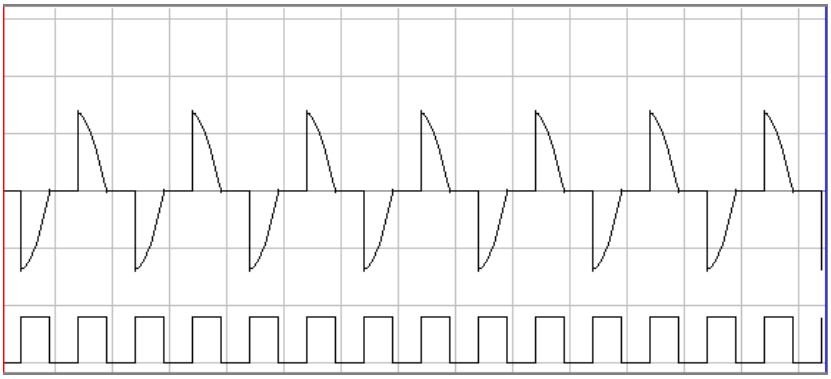
Gambar 4.18 Grafik Tegangan AC dengan PWM Sebesar 255

2. Ketika suhu mencapai 111^0C, PWM yang dihasilkan sebesar 190



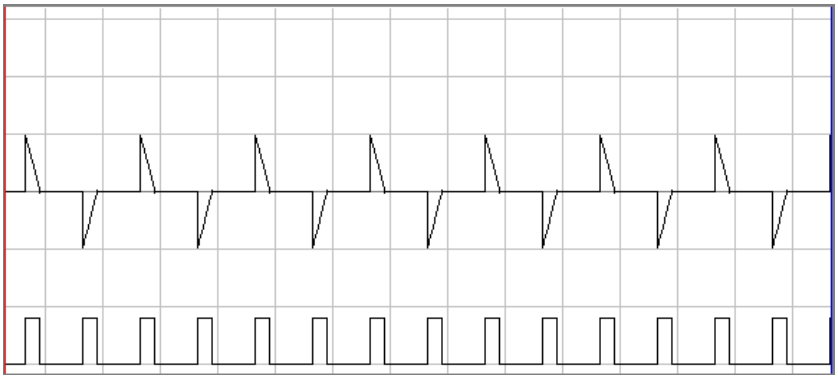
Gambar 4.19 Grafik Tegangan AC dengan PWM Sebesar 190

3. Ketika suhu mencapai 114^0C, PWM yang dihasilkan sebesar 128



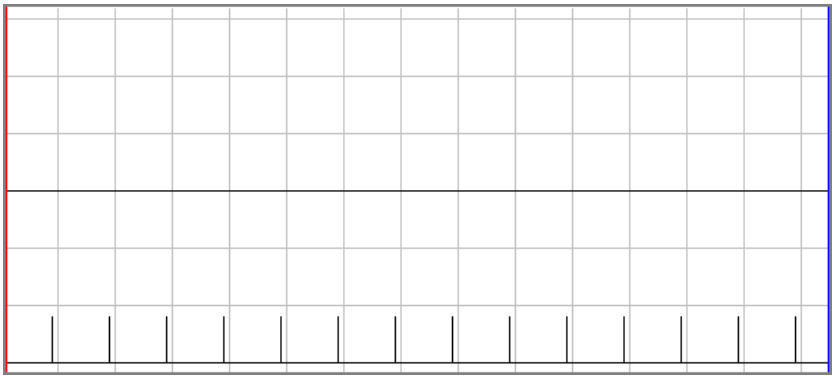
Gambar 4.20 Grafik Tegangan AC dengan PWM Sebesar 128

4. Ketika suhu mencapai 117^0C, PWM yang dihasilkan sebesar 64



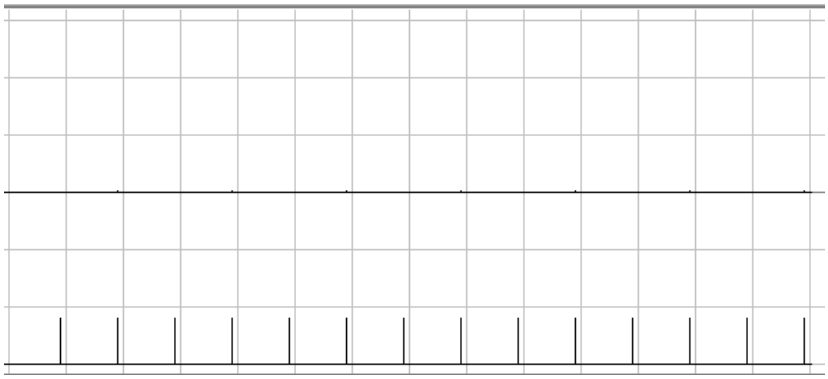
Gambar 4.21 Grafik Tegangan AC dengan PWM Sebesar 64

5. Ketika suhu mencapai 118.8^{0}C, PWM yang dihasilkan sebesar 25



Gambar 4.22 Grafik Tegangan AC dengan PWM sebesar 25

6. Ketik suhu mencapai 120^{0}C dan >120^{0}C, PWM yang dihasilkan sebesar 0



Gambar 4.23 Grafik Tegangan AC dengan PWM sebesar 0

Selanjutnya untuk jarak komunikasi telemetri antara *smartphone Android* dengan Arduino Mega hanya sebatas jarak maksimal 10 meter, sehingga untuk penerapan di industri tidak tepat. Untuk itu perlu mengganti perangkat *Bluetooth* dengan perangkat yang memiliki jangkauan komunikasi yang lebih jauh lagi seperti Wifi atau Internet.

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Setelah melakukan perancangan, pengujian, implementasi sistem, dan analisa, dapat diambil kesimpulan sebagai berikut:

1. Sistem kendali suhu otomatis untuk proses sterilisasi susu yang telah dibuat dapat digunakan untuk menjaga kestabilan suhu pada saat proses sterilisasi susu dilakukan, yaitu pada suhu 120^0C selama 15 menit dengan persentasi *error* sebesar 0,24% yang menggunakan parameter KP =7, KI = 7, dan KD = 40 yang dapat dilihat pada percobaan ke-4.
2. Sistem kendali suhu otomatis untuk proses sterilisasi susu yang telah dibuat dapat dipantau dan dikontrol dari jarak jauh (telemetri) dengan menggunakan *smartphone* Android melalui koneksi *Bluetooth* dengan radius maksimal sebesar 10 meter.

5.2 Saran

Berikut adalah beberapa saran yang ditujukan untuk pembaca tulisan dari penelitian ini agar alat ini bisa dikembangkan menjadi lebih baik lagi, antara lain:

1. Pada sistem kendali suhu otomatis yang telah dibuat perlu ditambahkan kontrol logika fuzzy (*fuzzy logic*) mendampingi kontrol PID (*Proportional, Integral, Derivative)* agar sistem kontrol suhu otomatis dapat menjaga kestabilan suhu untuk proses sterilisasi susu dengan tingkat kesalahan lebih mendekati nol (0).
2. Pada sistem kendali suhu otomatis yang telah dibuat perlu mengganti perangkat telemetri *Bluetooth* dengan perangkat telemetri yang memiliki jangkauan koneksi yang lebih jauh seperti wifi dan internet.

DAFTAR PUSTAKA

Budiharto, W., & Rizal, G. (2007). 12 Proyek Mikrokontroler Untuk Pemula. In *Elektronika & Pemrograman* (p. 213).

Chavan, R. S., Chavan, S. R., Khedkar, C. D., & Jana, A. H. (2011). UHT milk processing and effect of plasmin activity on shelf life: A review. *Comprehensive Reviews in Food Science and Food Safety, 10*(5), 251–268. http://doi.org/10.1111/j.1541-4337.2011.00157.x

Davis, D. (2002). Bluetooth. *Network Security*, *2002*(4), 11–12. http://doi.org/10.1016/S1353-4858(02)00413-0

Mellis, D. (2011). Arduino Mega 2560. *Retrieved November*, 2560.

No, B. (1998). *Wildlife Radio-telemetry*. *Components* (Vol. 5). http://doi.org/0772635358

Pusdatin. (2013). Susu. *Buletin Konsumsi Pangan, 4*(4), 35–45.

Red, I. (2006). *MLX90614 - Single and Dual Zone Infrared Thermometer in TO-39*. *Melexis*.

Semiconductors, V. (n.d.). H11AA1 Vishay Semiconductors Optocoupler , Phototransistor *Output* , AC Input , with Base Connection H11AA1, 1–7.

Texas Instruments. (1998). Moc3020 thru moc3023 optocouplers/optoisolators, (October 1986), 1–7.

Arduino. (2015, Oktober 3). *Arduino*. Retrieved from Arduino: http://arduino.cc

Charles L. Philips, R. H. (1997). *Feedback Control System.* Yogyakarta: Andi.

Heru Dibyo Laksono, M. (2014). *Sistem Kendali dengan Matlab.* Yogyakarta: Graha Ilmu.

Istiany, A., Yusro, M., Nasution, N., Amalia, R., & Muksin, &. (2009). *Buku Pedoman Skripsi/Komprehensif/Karya Inovatif (S1).* Jakarta: Universitas Negeri Jakarta.

Katsuhiko, & Ogata. (1997). *Teknik Kontrol Automatik Jilid 1 Edisi 2.* Jakarta: Erlangga.

kbbi. (20015, November 25). *sterilisasi*. Retrieved from kbbi: kbbi.web.id/sterilisasi

KBBI. (2015, Oktober 3). *Sterilisasi*. Retrieved from KBBI: kbbi.web.id/sterilisasi

Petruzella, F. D. (2001). *Elektronik Industri.* Yogyakarta: Andi.

Purnama, A. (2015, December 21). *Low Pass Filter*. Retrieved from Elektronika Dasar: http://elektronika-dasar.web.id/low-pass-filter-lpf-rc/

Susanti. (2015, Oktober 3). *Sterilisasi Susu dengan Metode Konvensional.* Retrieved from Binus: http://library.binus.ac.id/eColls/eThesisdoc/Bab2HTML/2007300476TIASBAB2/page29.htm

Syahwil, M. (2013). *Panduan Mudah Simulasi & Praktek Mikrokontroler Arduino.* Yogyakarta: Andi.

LAMPIRAN

1. Baris Program Arduino Mega 2560

```
/*=========================================================*/
#include <SoftwareSerial.h>
SoftwareSerial BT(52, 50); // RX,TX
/*
The circuit:
 * RX is digital pin 52 (connect to TX Bluetooth HC-06)
 * TX is digital pin 50 (connect to RX Bluetooth HC-06)
*/
#include <i2cmaster.h>
#include <EEPROM.h>
#include <LiquidCrystal.h>
LiquidCrystal lcd(31, 33, 35, 37, 39, 41);
/*=========================================================*/
/* Membuat kustom karakter */
byte simbolDerajat[8] = {
  0b01110,
  0b01010,
  0b01110,
  0b00000,
  0b00000,
  0b00000,
  0b00000,
  0b00000
};
byte simbolAtas[8] = {
  0b00100,
  0b01110,
  0b11111,
  0b00100,
  0b00100,
  0b00100,
  0b00100,
  0b00100
};
byte simbolBawah[8] = {
  0b00100,
  0b00100,
  0b00100,
  0b00100,
  0b00100,
  0b11111,
  0b01110,
  0b00100
};
byte simbolGaris[8] = {
  0b00100,
  0b00100,
  0b00100,
  0b00100,
  0b00100,
  0b00100,
  0b00100,
  0b00100
};
byte simbolTitik[8] = {
  0b00000,
  0b00000,
  0b01010,
  0b10101,
```

```
  0b01010,
  0b00000,
  0b00000,
  0b00000
};
byte simbolLoad[8] = {
  0b00000,
  0b00000,
  0b00000,
  0b00000,
  0b01010,
  0b10101,
  0b01010,
  0b00000
};
/*============================================================*/
int
hour, m, s, sampling, ts, menu, suhu0, kecepatan, a, pwm, pwmout,
rise_time, jumlah_cuplik, kp, ki, kd, jam, menit, detik, inByte, KP, KI,
KD, KECEPATAN;

float
DETIK, RISE_TIME, suhu, cos_a, V, error, last_error, pid, p, i, d,
data_suhu, suhu_rerata, persentase;

boolean
OK, UP, DOWN, CANCEL;

const int
okPin = 43, upPin = 45, downPin = 47, cancelPin = 49, heaterPin = 13,
lm35Pin = A0;

String
TIME, spasi, spasi2, titik, derajat, strpwm, strsuhu, strtime,
strtegangan, AC;

unsigned long
pmilis, cmilis;
/*============================================================*/
void setup() {
  titik        = ":";
  spasi        = "  ";
  spasi2       = "                ";
  strpwm       = "PWM  : ";
  strtegangan  = "Vrms : ";
  strsuhu      = "Temp : ";
  strtime      = "Time : ";
  AC           = "V  ";
  /*============================================================*/
  i2c_init(); // Initialise the i2c bus
  PORTC = (1 << PORTC4) | (1 << PORTC5);
  /*------------------------------------------------------------*/
  // Bluetooth with baudrate=9600bps
  BT.begin(9600);
  // LCD 16x4
  lcd.begin(16, 4);
  // Create New Char
  lcd.createChar(1, simbolDerajat);
  lcd.createChar(2, simbolAtas);
  lcd.createChar(3, simbolBawah);
  lcd.createChar(4, simbolGaris);
  lcd.createChar(5, simbolTitik);
  lcd.createChar(6, simbolLoad);
```

```
/*=========================================================*/
/*Reset parameter*/
a = 0;
menu = 0;
data_suhu = 0;
jam = 0;
menit = 0;
detik = 0;
DETIK = 0;
pmilis = 0;
cmilis = 0;
jumlah_cuplik = 0;
RISE_TIME = 0;
rise_time = 0;
s = 5;
/*=========================================================*/
pinMode(heaterPin, OUTPUT);
pinMode(okPin, INPUT); pinMode(cancelPin, INPUT);
pinMode(upPin, INPUT); pinMode(downPin, INPUT);
/*=========================================================*/
lets_begin:
lcd.clear ();
judul_awal();
delay(800);
judul();
lcd.setCursor (0, 1); lcd.print("Sistem Kontrol  ");
lcd.setCursor (0, 2); lcd.print("Suhu Otomatis   ");
delay(2000);
/*=========================================================*/
pilih:
judul3();
tombol();
menu = menu;
inByte = BT.read();
if (inByte == 'b') {
  goto mulai;
}
if (UP == HIGH) {
  delay(250);
  menu++;
}
else if (DOWN == HIGH) {
  delay(250);
  menu--;
}
if (menu > 1) {
  menu = 0;
}
else if (menu < 0) {
  menu = 1;
}
switch (menu) {
  case 0 :
    lcd.setCursor (0, 0); lcd.print(" Silahkan Pilih ");
    lcd.setCursor (0, 2); lcd.print("  SetParameter  ");
    tombol();
    if (OK == HIGH) {
      delay(250);
      lcd.clear();
      goto set_parameter;
    }
    else if (CANCEL == HIGH) {
      delay(250);
      goto lets_begin;
    }
```

```
    else goto pilih;
    break;
  case 1 :
    lcd.setCursor (0, 0); lcd.print(" Silahkan Pilih ");
    lcd.setCursor (0, 2); lcd.print("      Mulai     ");
    tombol();
    if (OK == HIGH) {
      delay(250);
      goto mulai;
    }
    else if (CANCEL == HIGH) {
      delay(250);
      goto lets_begin;
    }
    else goto pilih;
    break;
}
/*=========================================================*/
set_parameter:
baca();
judul2();
/*=========================================================*/
set_suhu:
judul2();
tombol();
suhu0 = suhu0;
if (UP == HIGH) {
  delay(250);
  suhu0 = suhu0 + 5;
}
else if (DOWN == HIGH) {
  delay(250);
  suhu0 = suhu0 - 5;
}
if (suhu0 > 200) {
  suhu0 = 0;
}
else if (suhu0 < 0) {
  suhu0 = 200;
}

lcd.setCursor(0, 2); lcd.print("  Suhu = "); lcd.print(suhu0);
lcd.write(1); lcd.print("C ");

if (OK == HIGH) {
  delay(250);
  lcd.setCursor(0, 2); lcd.print(spasi2);
  goto set_waktu;
}
else if (CANCEL == HIGH) {
  delay(250);
  lcd.setCursor(0, 2); lcd.print(spasi2);
  goto pilih;
}
else goto set_suhu;
/*=========================================================*/
set_waktu:
tombol();
hour = hour;
if (UP == HIGH) {
  delay(250);
  hour++;
}
else if (DOWN == HIGH) {
  delay(250);
```

```
  hour--;
}
if (hour > 30) {
  hour = 0;
}
else if (hour < 0) {
  hour = 30;
}

lcd.setCursor(0, 2); lcd.print(" Time = "); lcd.print(hour);
lcd.print(" menit  ");

if (OK == HIGH) {
  delay(250);
  lcd.setCursor(0, 2); lcd.print(spasi2);
  goto set_pwm_awal;
}
else if (CANCEL == HIGH) {
  delay(250);
  lcd.setCursor(0, 2); lcd.print(spasi2);
  goto set_suhu;
}
else goto set_waktu;
/*=============================================================*/
set_pwm_awal:
tombol();
kecepatan = kecepatan;
if (UP == HIGH) {
  delay(250);
  kecepatan = kecepatan + 5;
}
else if (DOWN == HIGH) {
  delay(250);
  kecepatan = kecepatan - 5;
}
if (kecepatan > 255) {
  kecepatan = 0;
}
else if (kecepatan < 0) {
  kecepatan = 255;
}

lcd.setCursor(0, 2); lcd.print(" PWMawal="); lcd.print(kecepatan);
lcd.print("/255  ");

if (OK == HIGH) {
  delay(250);
  lcd.setCursor(0, 2); lcd.print(spasi2);
  goto set_time_sampling;
}
else if (CANCEL == HIGH) {
  delay(250);
  lcd.setCursor(0, 2); lcd.print(spasi2);
  goto set_waktu;
}
else goto set_pwm_awal;
/*=============================================================*/
set_time_sampling:
tombol();
ts = ts;
sampling = sampling;
if (UP == HIGH) {
  delay(250);
  sampling = sampling + 1;
}
```

```
else if (DOWN == HIGH) {
  delay(250);
  sampling = sampling - 1;
}
if (sampling < 0) {
  sampling = 5;
}
else if (sampling > 5) {
  sampling = 0;
}

switch (sampling) {
  case 0 : ts = 0; break;
  case 1 : ts = 100; break;
  case 2 : ts = 200; break;
  case 3 : ts = 250; break;
  case 4 : ts = 500; break;
  case 5 : ts = 1000; break;
}

lcd.setCursor(0, 2); lcd.print("  Ts = "); lcd.print(ts);
lcd.print(" ms ");

if (OK == HIGH) {
  delay(250);
  lcd.setCursor(0, 2); lcd.print(spasi2);
  goto set_KP;
}
else if (CANCEL == HIGH) {
  delay(250);
  lcd.setCursor(0, 2); lcd.print(spasi2);
  goto set_pwm_awal;
}
else goto set_time_sampling;
/*=============================================================*/
set_KP:
tombol();
kp = kp;
if (UP == HIGH) {
  delay(250);
  kp++;
}
else if (DOWN == HIGH) {
  delay(250);
  kp--;
}
if (kp > 255) {
  kp = 0;
}
else if (kp < 0) {
  kp = 255;
}

lcd.setCursor(0, 2); lcd.print(" Nilai KP = "); lcd.print(kp);
lcd.print(spasi);

if (OK == HIGH) {
  delay(250);
  lcd.setCursor(0, 2);
  lcd.print(spasi2);
  goto set_KI;
}
else if (CANCEL == HIGH) {
  delay(250);
  lcd.setCursor(0, 2); lcd.print(spasi2);
```

```
  goto set_time_sampling;
}
else goto set_KP;
/*=========================================================*/
set_KI:
tombol();
ki = ki;
if (UP == HIGH) {
  delay(250);
  ki++;
}
else if (DOWN == HIGH) {
  delay(250);
  ki--;
}
if (ki > 255) {
  ki = 0;
}
else if (ki < 0) {
  ki = 255;
}

lcd.setCursor(0, 2); lcd.print(" Nilai KI = "); lcd.print(ki);
lcd.print(spasi);

if (OK == HIGH) {
  delay(250);
  lcd.setCursor(0, 2); lcd.print(spasi2);
  goto set_KD;
}
else if (CANCEL == HIGH) {
  delay(250);
  lcd.setCursor(0, 2); lcd.print(spasi2);
  goto set_KP;
}
else goto set_KI;
/*=========================================================*/
set_KD:
tombol();
kd = kd;
if (UP == HIGH) {
  delay(250);
  kd++;
}
else if (DOWN == HIGH) {
  delay(250);
  kd--;
}

if (kd > 255) {
  kd = 0;
}
else if (kd < 0) {
  kd = 255;
}

lcd.setCursor(0, 2); lcd.print(" Nilai KD = "); lcd.print(kd);
lcd.print(spasi);

if (OK == HIGH) {
  delay(250);
  goto simpan;
}
else if (CANCEL == HIGH) {
  delay(250);
```

```
  lcd.setCursor(0, 2); lcd.print(spasi2);
  goto set_KI;
}
else goto set_KD;
/*============================================================*/
simpan:
tulis();
lcd.clear();
judul();
lcd.setCursor(0, 1); lcd.print(" Save Parameter ");
loading();
lcd.clear();
judul();
lcd.setCursor(0, 1);
for (int i=0; i<16; i++) {
  lcd.write(5);
}
save();
goto final;
/*============================================================*/
mulai:
baca();
lcd.clear();
judul();
lcd.setCursor(0, 1); lcd.print(" Load Parameter ");
loading();
lcd.clear();
judul();
lcd.setCursor(0, 1);
for (int i=0; i<16; i++) {
  lcd.write(5);
}
load();
goto final;
/*============================================================*/
final:
menu = 0;
lcd.clear();
akhir:
tombol();
menu = menu;
if (UP == HIGH) {
  delay(150);
  menu--;
}
else if (DOWN == HIGH) {
  delay(150);
  menu++;
}
if (menu < 0) {
  menu = 6;
}
else if (menu > 6) {
  menu = 0;
}

frame();
switch (menu) {
  case 0 :
    lcd.setCursor(3, 0);    lcd.print("Kp  : ");    lcd.print(kp);
    lcd.print(spasi);
    lcd.setCursor(3, 1);    lcd.print("Ki  : ");    lcd.print(ki);
    lcd.print(spasi);
    lcd.setCursor(3, 2);    lcd.print("Kd  : ");    lcd.print(kd);
    lcd.print(spasi);
```

```
    lcd.setCursor(3, 3);    lcd.print("time: ");    lcd.print(hour);
    lcd.print(spasi);
    break;
  case 1 :
    lcd.setCursor(3, 0);    lcd.print("Ki   : "); lcd.print(ki);
    lcd.print(spasi);
    lcd.setCursor(3, 1);    lcd.print("Kd   : "); lcd.print(kd);
    lcd.print(spasi);
    lcd.setCursor(3, 2);    lcd.print("time: "); lcd.print(hour);
    lcd.print(spasi);
    lcd.setCursor(3, 3);    lcd.print("temp: "); lcd.print(suhu0);
    lcd.print(spasi);
    break;
  case 2 :
    lcd.setCursor(3, 0);    lcd.print("Kd   : "); lcd.print(kd);
    lcd.print(spasi);
    lcd.setCursor(3, 1);    lcd.print("time: "); lcd.print(hour);
    lcd.print(spasi);
    lcd.setCursor(3, 2);    lcd.print("temp: "); lcd.print(suhu0);
    lcd.print(spasi);
    lcd.setCursor(3, 3);    lcd.print("pwm : "); lcd.print(kecepatan);
    lcd.print(spasi);
    break;
  case 3 :
    lcd.setCursor(3, 0);    lcd.print("time: "); lcd.print(hour);
    lcd.print(spasi);
    lcd.setCursor(3, 1);    lcd.print("temp: "); lcd.print(suhu0);
    lcd.print(spasi);
    lcd.setCursor(3, 2);    lcd.print("pwm : "); lcd.print(kecepatan);
    lcd.print(spasi);
    lcd.setCursor(3, 3);    lcd.print("ts   : "); lcd.print(ts);
    lcd.print(spasi);
    break;
  case 4 :
    lcd.setCursor(3, 0);    lcd.print("temp: "); lcd.print(suhu0);
    lcd.print(spasi);
    lcd.setCursor(3, 1);    lcd.print("pwm : "); lcd.print(kecepatan);
    lcd.print(spasi);
    lcd.setCursor(3, 2);    lcd.print("ts   : "); lcd.print(ts);
    lcd.print(spasi);
    lcd.setCursor(3, 3);    lcd.print("Kp   : "); lcd.print(kp);
    lcd.print(spasi);
    break;
  case 5 :
    lcd.setCursor(3, 0);    lcd.print("pwm : "); lcd.print(kecepatan);
    lcd.print(spasi);
    lcd.setCursor(3, 1);    lcd.print("ts   : "); lcd.print(ts);
    lcd.print(spasi);
    lcd.setCursor(3, 2);    lcd.print("Kp   : "); lcd.print(kp);
    lcd.print(spasi);
    lcd.setCursor(3, 3);    lcd.print("Ki   : "); lcd.print(ki);
    lcd.print(spasi);
    break;
  case 6 :
    lcd.setCursor(3, 0);    lcd.print("ts   : "); lcd.print(ts);
    lcd.print(spasi);
    lcd.setCursor(3, 1);    lcd.print("Kp   : "); lcd.print(kp);
    lcd.print(spasi);
    lcd.setCursor(3, 2);    lcd.print("Ki   : "); lcd.print(ki);
    lcd.print(spasi);
    lcd.setCursor(3, 3);    lcd.print("Kd   : "); lcd.print(kd);
    lcd.print(spasi);
    break;
}
```

```
  inByte = BT.read();
  if (inByte == 'c') {
    lcd.clear();
  }
  else if (OK == HIGH) {
    delay(250);
    lcd.clear();
  }
  else if (CANCEL == HIGH) {
    delay(250);
    lcd.setCursor(0, 2);
    lcd.print(spasi2);
    goto pilih;
  }
  else goto akhir;
}
/*=========================================================*/
void judul_awal() {
  lcd.setCursor (7, 0); lcd.print("--"); lcd.setCursor(7, 3);
  lcd.print("--"); delay(15);
  lcd.setCursor (6, 0); lcd.print("----"); lcd.setCursor(6, 3);
  lcd.print("----"); delay(15);
  lcd.setCursor (5, 0); lcd.print("------"); lcd.setCursor(5, 3);
  lcd.print("------"); delay(15);
  lcd.setCursor (4, 0); lcd.print("--------"); lcd.setCursor(4, 3);
  lcd.print("--------"); delay(15);
  lcd.setCursor (3, 0); lcd.print("----------"); lcd.setCursor(3, 3);
  lcd.print("----------"); delay(15);
  lcd.setCursor (2, 0); lcd.print("------------"); lcd.setCursor(2, 3);
  lcd.print("------------"); delay(15);
  lcd.setCursor (1, 0); lcd.print("--------------"); lcd.setCursor(1, 3);
  lcd.print("--------------"); delay(15);
  lcd.setCursor (0, 0); lcd.print("----------------");
  lcd.setCursor(0, 3); lcd.print("----------------"); delay(200);
  lcd.setCursor(0, 1); lcd.print("A"); delay(10);
  lcd.setCursor(1, 1); lcd.print("b"); delay(10);
  lcd.setCursor(2, 1); lcd.print("d"); lcd.setCursor(13, 2);
  lcd.print("r"); delay(10);
  lcd.setCursor(3, 1); lcd.print("u"); lcd.setCursor(12, 2);
  lcd.print("e"); delay(10);
  lcd.setCursor(4, 1); lcd.print("l"); lcd.setCursor(11, 2);
  lcd.print("e"); delay(10);
  lcd.setCursor(6, 1); lcd.print("R"); lcd.setCursor(10, 2);
  lcd.print("n"); delay(10);
  lcd.setCursor(7, 1); lcd.print("o"); lcd.setCursor(9, 2);
  lcd.print("i"); delay(10);
  lcd.setCursor(8, 1); lcd.print("z"); lcd.setCursor(8, 2);
  lcd.print("g"); delay(10);
  lcd.setCursor(9, 1); lcd.print("a"); lcd.setCursor(7, 2);
  lcd.print("n"); delay(10);
  lcd.setCursor(10, 1); lcd.print("k"); lcd.setCursor(6, 2);
  lcd.print("E"); delay(10);
  lcd.setCursor(4, 2); lcd.print("n"); delay(10);
  lcd.setCursor(3, 2); lcd.print("a"); delay(10);
  lcd.setCursor(1, 2); lcd.print("m"); delay(10);
  lcd.setCursor(0, 2); lcd.print("I"); delay(500);
  lcd.setCursor(0, 2); lcd.print("   an Engineer"); delay(10);
  lcd.setCursor(0, 2); lcd.print("I   n Engineer"); delay(10);
  lcd.setCursor(0, 2); lcd.print("Im    Engineer"); delay(10);
  lcd.setCursor(0, 2); lcd.print("Im a   ngineer"); delay(10);
  lcd.setCursor(0, 2); lcd.print("Im an   gineer"); delay(10);
  lcd.setCursor(0, 2); lcd.print("Im an E  ineer"); delay(10);
  lcd.setCursor(0, 2); lcd.print("Im an En  neer"); delay(10);
  lcd.setCursor(0, 2); lcd.print("Im an Eng  eer"); delay(10);
  lcd.setCursor(0, 2); lcd.print("Im an Engi  er"); delay(10);
```

```
  lcd.setCursor(0, 2); lcd.print("Im an Engin  r"); delay(10);
  lcd.setCursor(0, 2); lcd.print("Im an Engine  "); delay(10);
  lcd.setCursor(0, 2); lcd.print("Im an Enginee "); delay(10);
  lcd.setCursor(0, 2); lcd.print("Im an Engineer"); delay(300);
  lcd.setCursor(0, 1); lcd.print("Abdul Roz  "); delay(10);
  lcd.setCursor(0, 1); lcd.print("Abdul Ro  k"); delay(10);
  lcd.setCursor(0, 1); lcd.print("Abdul R  ak"); delay(10);
  lcd.setCursor(0, 1); lcd.print("Abdul   zak"); delay(10);
  lcd.setCursor(0, 1); lcd.print("Abdu   ozak"); delay(10);
  lcd.setCursor(0, 1); lcd.print("Abd   Rozak"); delay(10);
  lcd.setCursor(0, 1); lcd.print("Ab  l Rozak"); delay(10);
  lcd.setCursor(0, 1); lcd.print("A  ul Rozak"); delay(10);
  lcd.setCursor(0, 1); lcd.print("  dul Rozak"); delay(10);
  lcd.setCursor(0, 1); lcd.print(" bdul Rozak"); delay(10);
  lcd.setCursor(0, 1); lcd.print("Abdul Rozak"); delay(300);
  lcd.setCursor(0, 1); lcd.print("  dul Rozak"); lcd.setCursor(0, 2);
  lcd.print("   an Engineer"); delay(10);
  lcd.setCursor(0, 1); lcd.print("A  ul Rozak"); lcd.setCursor(0, 2);
  lcd.print("I   n Engineer"); delay(10);
  lcd.setCursor(0, 1); lcd.print("Ab  l Rozak"); lcd.setCursor(0, 2);
  lcd.print("Im    Engineer"); delay(10);
  lcd.setCursor(0, 1); lcd.print("Abd   Rozak"); lcd.setCursor(0, 2);
  lcd.print("Im a   ngineer"); delay(10);
  lcd.setCursor(0, 1); lcd.print("Abdu   ozak"); lcd.setCursor(0, 2);
  lcd.print("Im an   gineer"); delay(10);
  lcd.setCursor(0, 1); lcd.print("Abdul   zak"); lcd.setCursor(0, 2);
  lcd.print("Im an E  ineer"); delay(10);
  lcd.setCursor(0, 1); lcd.print("Abdul R  ak"); lcd.setCursor(0, 2);
  lcd.print("Im an En  neer"); delay(10);
  lcd.setCursor(0, 1); lcd.print("Abdul Ro  k"); lcd.setCursor(0, 2);
  lcd.print("Im an Eng  eer"); delay(10);
  lcd.setCursor(0, 1); lcd.print("Abdul Roz  "); lcd.setCursor(0, 2);
  lcd.print("Im an Engi  er"); delay(10);
  lcd.setCursor(0, 1); lcd.print("Abdul Roza "); lcd.setCursor(0, 2);
  lcd.print("Im an Engin  r"); delay(10);
  lcd.setCursor(0, 1); lcd.print("Abdul Rozak"); lcd.setCursor(0, 2);
  lcd.print("Im an Engine  "); delay(10);
  lcd.setCursor(0, 2); lcd.print("Im an Enginee "); delay(10);
  lcd.setCursor(0, 2); lcd.print("Im an Engineer"); delay(500);
}
/*============================================================*/
void judul() {
  lcd.setCursor (0, 0); lcd.print("----------------");
  lcd.setCursor (0, 3); lcd.print("----------------");
}
/*============================================================*/
void judul2() {
  lcd.setCursor (0, 0); lcd.print(" SET PARAMETER: ");
  lcd.setCursor (0, 1); lcd.print("----------------");
  lcd.setCursor (0, 3); lcd.print("----------------");
}
/*============================================================*/
void judul3() {
  lcd.setCursor (0, 1); lcd.print("----------------");
  lcd.setCursor (0, 3); lcd.print("----------------");
}
/*============================================================*/
void frame() {
  lcd.setCursor(0, 0); lcd.write(2); lcd.setCursor(15, 0); lcd.write(2);
  lcd.setCursor(0, 1); lcd.write(4); lcd.setCursor(15, 1); lcd.write(4);
  lcd.setCursor(0, 2); lcd.write(4); lcd.setCursor(15, 2); lcd.write(4);;
  lcd.setCursor(0, 3); lcd.write(3); lcd.setCursor(15, 3); lcd.write(3);
}
/*============================================================*/
void loading() {
```

```
  lcd.setCursor(0, 2); lcd.write(6); delay(20);
  lcd.setCursor(1, 2); lcd.write(6); delay(20);
  lcd.setCursor(2, 2); lcd.write(6); delay(20);
  lcd.setCursor(3, 2); lcd.write(6); delay(20);
  lcd.setCursor(4, 2); lcd.write(6); delay(20);
  lcd.setCursor(5, 2); lcd.write(6); delay(20);
  lcd.setCursor(6, 2); lcd.write(6); delay(20);
  lcd.setCursor(7, 2); lcd.write(6); delay(20);
  lcd.setCursor(8, 2); lcd.write(6); delay(20);
  lcd.setCursor(9, 2); lcd.write(6); delay(20);
  lcd.setCursor(10, 2); lcd.write(6); delay(20);
  lcd.setCursor(11, 2); lcd.write(6); delay(20);
  lcd.setCursor(12, 2); lcd.write(6); delay(20);
  lcd.setCursor(13, 2); lcd.write(6); delay(20);
  lcd.setCursor(14, 2); lcd.write(6); delay(20);
  lcd.setCursor(15, 2); lcd.write(6); delay(500);
}
/*=========================================================*/
void load() {
  lcd.setCursor(0, 2); lcd.print("Load succesfully"); delay(500);
  lcd.setCursor(0, 2); lcd.print("  ad succesfully"); delay(20);
  lcd.setCursor(0, 2); lcd.print("L  d succesfully"); delay(20);
  lcd.setCursor(0, 2); lcd.print("Lo   succesfully"); delay(20);
  lcd.setCursor(0, 2); lcd.print("Loa   uccesfully"); delay(20);
  lcd.setCursor(0, 2); lcd.print("Load   ccesfully"); delay(20);
  lcd.setCursor(0, 2); lcd.print("Load s  cesfully"); delay(20);
  lcd.setCursor(0, 2); lcd.print("Load su  esfully"); delay(20);
  lcd.setCursor(0, 2); lcd.print("Load suc  sfully"); delay(20);
  lcd.setCursor(0, 2); lcd.print("Load succ  fully"); delay(20);
  lcd.setCursor(0, 2); lcd.print("Load succe  ully"); delay(20);
  lcd.setCursor(0, 2); lcd.print("Load succes  lly"); delay(20);
  lcd.setCursor(0, 2); lcd.print("Load succesf  ly"); delay(20);
  lcd.setCursor(0, 2); lcd.print("Load succesfu  y"); delay(20);
  lcd.setCursor(0, 2); lcd.print("Load succesful  "); delay(20);
  lcd.setCursor(0, 2); lcd.print("Load succesfull "); delay(20);
  lcd.setCursor(0, 2); lcd.print("Load succesfully"); delay(500);
}
void save() {
  lcd.setCursor(0, 2); lcd.print("Write succesfull"); delay(500);
  lcd.setCursor(0, 2); lcd.print("  ite succesfull"); delay(20);
  lcd.setCursor(0, 2); lcd.print("W  te succesfull"); delay(20);
  lcd.setCursor(0, 2); lcd.print("Wr  e succesfull"); delay(20);
  lcd.setCursor(0, 2); lcd.print("Wri   succesfull"); delay(20);
  lcd.setCursor(0, 2); lcd.print("Writ   uccesfull"); delay(20);
  lcd.setCursor(0, 2); lcd.print("Write   ccesfull"); delay(20);
  lcd.setCursor(0, 2); lcd.print("Write s  cesfull"); delay(20);
  lcd.setCursor(0, 2); lcd.print("Write su  esfull"); delay(20);
  lcd.setCursor(0, 2); lcd.print("Write suc  sfull"); delay(20);
  lcd.setCursor(0, 2); lcd.print("Write succ  full"); delay(20);
  lcd.setCursor(0, 2); lcd.print("Write succe  ull"); delay(20);
  lcd.setCursor(0, 2); lcd.print("Write succes  ll"); delay(20);
  lcd.setCursor(0, 2); lcd.print("Write succesf  l"); delay(20);
  lcd.setCursor(0, 2); lcd.print("Write succesfu  "); delay(20);
  lcd.setCursor(0, 2); lcd.print("Write succesful "); delay(20);
  lcd.setCursor(0, 2); lcd.print("Write succesfull"); delay(500);
}
/*=========================================================*/
void angle() {
  switch (pwm) {
    case 255 : cos_a = 1; break;
    case 254 : cos_a = 0.99992411011483056875185094912157; break;
    case 253 : cos_a = 0.99969645197787161706632560365192; break;
    case 252 : cos_a = 0.99931706014302288834641482622583; break;
    case 251 : cos_a = 0.99878599219428994437181274925287; break;
```

```
case 250 : cos_a = 0.99810332873704407815955807227985; break;
case 249 : cos_a = 0.99726917338578804922110119591952; break;
case 248 : cos_a = 0.99628365274842949812969013955386; break;
case 247 : cos_a = 0.99514691640706442737560041937711; break;
case 246 : cos_a = 0.99385913689527366518807139262033; break;
case 245 : cos_a = 0.99242050967193575826145605410729; break;
case 244 : cos_a = 0.99083125309156026805871214960483; break;
case 243 : cos_a = 0.98909160837114597349770837305954; break;
case 242 : cos_a = 0.98720183955356901027473053727675; break;
case 241 : cos_a = 0.98516223346750650376428747329 4; break;

case 240 : cos_a = 0.98297306839017782819488448552; break;
case 239 : cos_a = 0.98063477046897775039594182001107; break;
case 238 : cos_a = 0.97814760073380563792856674786 96; break;
case 237 : cos_a = 0.97551196798043663907250951826242; break;
case 236 : cos_a = 0.97272827224460475775823576547705; break;
case 235 : cos_a = 0.96979693603500947181953601565396; break;
case 234 : cos_a = 0.96671840426918745962060860134947; break;
case 233 : cos_a = 0.96349314420598311851928511241106; break;
case 232 : cos_a = 0.96012164537462812477559713726104; break;
case 231 : cos_a = 0.95660441950044071937937033852; break;

case 230 : cos_a = 0.95294200042715655583102830341526; break;
case 229 : cos_a = 0.94913494403590122243937282478935; break;
case 228 : cos_a = 0.94518382816081953085197301472101; break;
case 227 : cos_a = 0.94108925250137158320169781292909; break;
case 226 : cos_a = 0.93685183853131060559449779220188; break;
case 225 : cos_a = 0.93247222940435580457311589182156; break;
case 224 : cos_a = 0.92795108985657464433233333102751; break;
case 223 : cos_a = 0.92328910610548935401844752313264; break;
case 222 : cos_a = 0.91848698574592298923718259 0606; break;
case 221 : cos_a = 0.91354545764260089550212757198532; break;

case 220 : cos_a = 0.90846527181952368611150364758592; break;
case 219 : cos_a = 0.90324713461288771244832696 1716; break;
case 218 : cos_a = 0.89789203222025809194535788094013; break;
case 217 : cos_a = 0.89240058324794782146825205472 03; break;
case 216 : cos_a = 0.88677368592006191052849449457395; break;
case 215 : cos_a = 0.88101219428578450600870881792559; break;
case 214 : cos_a = 0.87511698282226678109068266769; break;
case 213 : cos_a = 0.86908894630552831750205472549 3; break;
case 212 : cos_a = 0.86292896673896701213239468735 5; break;
case 211 : cos_a = 0.85663807786386276197714064058715; break;

case 210 : cos_a = 0.85021713572961415213414392294935; break;
case 209 : cos_a = 0.84366714783376633597193878088 32; break;
case 208 : cos_a = 0.83698910833197786760126642550977; break;
case 207 : cos_a = 0.83018403081555064232758076312608; break;
case 206 : cos_a = 0.82325294815758724163191029054514; break;
case 205 : cos_a = 0.81619691235622169087185254043141; break;
case 204 : cos_a = 0.80901643749474241022934171828 2; break;
case 203 : cos_a = 0.80171428398006669100068047335154; break;
case 202 : cos_a = 0.79428988957528607778853545451847; break;
case 201 : cos_a = 0.78674493803348324720305366463371; break;

case 200 : cos_a = 0.77908057452567043192436062060749; break;
case 199 : cos_a = 0.77129796234718064134011316848224; break;
case 198 : cos_a = 0.76339828274110296306506405656019; break;
case 197 : cos_a = 0.75538273471837581649752764794 8; break;
case 196 : cos_a = 0.74725253487889096249849341582542; break;
case 195 : cos_a = 0.73900891722065911592453430987265; break;
case 194 : cos_a = 0.73065313295869314626423582497762; break;
case 193 : cos_a = 0.72218645033200933579413436640283; break;
case 192 : cos_a = 0.71361015441175229462005605158471; break;
case 191 : cos_a = 0.70492554690614715747909569218667; break;
```

```
case 190 : cos_a = 0.69613394596292660828045808021714; break;
case 189 : cos_a = 0.68723668596926271660019350384 7; break;
case 188 : cos_a = 0.67823511734923397598988459667025; break;
case 187 : cos_a = 0.66913060635885821382627333068678; break;
case 186 : cos_a = 0.65245348787226003578846769 4186; break;
case 185 : cos_a = 0.65061830020424211372006253388 21; break;
case 184 : cos_a = 0.64121331483357836611985120863578; break;
case 183 : cos_a = 0.63171100625325095726242914957516; break;
case 182 : cos_a = 0.62211281672147398982444435568003; break;
case 181 : cos_a = 0.61242020304924910620681368730 2; break;

case 180 : cos_a = 0.60263463637925638917858815498684; break;
case 179 : cos_a = 0.59275760196255488503484428350 7; break;
case 178 : cos_a = 0.58279059893316091907009636382779; break;
case 177 : cos_a = 0.57273514008050521500300128023 29; break;
case 176 : cos_a = 0.56259275161982309561360391756483; break;
case 175 : cos_a = 0.55236497296050581076310052290037; break;
case 174 : cos_a = 0.54205335647244934834795081083673; break;
case 173 : cos_a = 0.53165946725043608800587117791977; break;
case 172 : cos_a = 0.52118488287658507448770131246566; break;
case 171 : cos_a = 0.5106311931809069659380958774 30; break;

case 170 : cos_a = 0.5; break;
case 169 : cos_a = 0.48929291693392360281375507169058; break;
case 168 : cos_a = 0.47851156910128654257862429118626; break;
case 167 : cos_a = 0.46765759289258680034054364830606; break;
case 166 : cos_a = 0.45673263572184059602386193841614; break;
case 165 : cos_a = 0.44573835577653826739645754937949; break;
case 164 : cos_a = 0.43467642176596495360749727835469; break;
case 163 : cos_a = 0.42354851266792428312668885930569; break;
case 162 : cos_a = 0.41235631747390350830550405554331; break;
case 161 : cos_a = 0.40110153493271876533772294978527; break;

case 160 : cos_a = 0.38978587329267936908286781204 5; break;
case 159 : cos_a = 0.37841105004231027666440127318 1; break;
case 158 : cos_a = 0.36697879164967207525326481625924; break;
case 157 : cos_a = 0.35549083330031805301230138770159; break;
case 156 : cos_a = 0.34394891863392813764514166609715; break;
case 155 : cos_a = 0.33235474796596645618863109731; break;
case 154 : cos_a = 0.32071023559025515003805714306922; break;
case 153 : cos_a = 0.30901643749474241022934171828 2; break;
case 152 : cos_a = 0.29727685063120266308262492159217; break;
case 151 : cos_a = 0.28549158627534203779223383043858; break;

case 150 : cos_a = 0.27366200720828635390779354368 1; break;
case 149 : cos_a = 0.26179285736304030214151290916281; break;
case 148 : cos_a = 0.24988298979423082382290608263 5; break;
case 147 : cos_a = 0.23793519504261878898146277085821; break;
case 146 : cos_a = 0.22595128654174765292955787840758; break;
case 145 : cos_a = 0.21393308320649743064933542 61; break;
case 144 : cos_a = 0.20188240915701025404079408963 93; break;
case 143 : cos_a = 0.18980109344182577268677382415 3; break;
case 142 : cos_a = 0.17769096976026862013663375636 89; break;
case 141 : cos_a = 0.16555387618412642543846237196 4; break;

case 140 : cos_a = 0.15339165487868537264875527121407; break;
case 139 : cos_a = 0.14120615182309139612027943811828; break;
case 138 : cos_a = 0.12892165302032762212068614 17; break;
case 137 : cos_a = 0.11677270176585630889169135255451; break;
case 136 : cos_a = 0.10452846326765347138341548025; break;
case 135 : cos_a = 0.09226835946330152396511 07715451; break;
case 134 : cos_a = 0.07942511885416354925729790 7202; break;
case 133 : cos_a = 0.06770800140470744961777711781405; break;
case 132 : cos_a = 0.05541147491596538669856292 1052; break;
```

```
case 131 : cos_a = 0.043106538086295574556555571369075; break;

case 130 : cos_a = 0.030795058556170353874564894976 24; break;
case 129 : cos_a = 0.018478904959120583376618608974; break;
case 128 : cos_a = 0.006154663813864738859660038138; break;
case 127 : cos_a = -0.006154663813864738859660038138; break;
case 126 : cos_a = -0.018478904959120583376618608974; break;
case 125 : cos_a = -0.030795058556170353874564894976 24; break;
case 124 : cos_a = -0.043106538086295574556555571369075; break;
case 123 : cos_a = -0.055411474915965386698562921052; break;
case 122 : cos_a = -0.067708001404707449617777111781405; break;
case 121 : cos_a = -0.079425118854163549257297907202; break;

case 120 : cos_a = -0.092268359463301523965110715451; break;
case 119 : cos_a = -0.104528463267653471383415480 25; break;
case 118 : cos_a = -0.116772701765856308891691351 25451; break;
case 117 : cos_a = -0.128921653020327622120686141 7; break;
case 116 : cos_a = -0.141206151823091396120279438 11828; break;
case 115 : cos_a = -0.153391654878685372648755271 21407; break;
case 114 : cos_a = -0.165553876184126425438462371964; break;
case 113 : cos_a = -0.177690969760268620136633756 3689; break;
case 112 : cos_a = -0.189801093441825772686773824153; break;
case 111 : cos_a = -0.201882409157010254087940896393; break;

case 110 : cos_a = -0.213933083206497430649335426 1; break;
case 109 : cos_a = -0.225951286541747652929557878407 58; break;
case 108 : cos_a = -0.237935195042618788981462770858 21; break;
case 107 : cos_a = -0.249882989794230823822906082635; break;
case 106 : cos_a = -0.261792857363040302141512909162 81; break;
case 105 : cos_a = -0.273662007208286353907793543681; break;
case 104 : cos_a = -0.285491586275342037792233830438 58; break;
case 103 : cos_a = -0.297276850631202663082624921592 17; break;
case 102 : cos_a = -0.309016437494742410229341718282; break;
case 101 : cos_a = -0.320710235590255150038057143069 22; break;

case 100 : cos_a = -0.332354747965966456188631097 31; break;
case 99 : cos_a = -0.343948918633928137645141666097 15; break;
case 98 : cos_a = -0.355490833300318053012301387701 59; break;
case 97 : cos_a = -0.366978791649672075253264816259 24; break;
case 96 : cos_a = -0.378411050042310276664401273181; break;
case 95 : cos_a = -0.389785873292679369082867812045; break;
case 94 : cos_a = -0.401101534932718765337722949785 27; break;
case 93 : cos_a = -0.412356317473903508305504055543 31; break;
case 92 : cos_a = -0.423548512667924283126688859305 69; break;
case 91 : cos_a = -0.434676421765964953607497278354 69; break;

case 90 : cos_a = -0.445738355776538267396457549379 49; break;
case 89 : cos_a = -0.456732635721840596023861938416 14; break;
case 88 : cos_a = -0.467657592892586800340543648306 06; break;
case 87 : cos_a = -0.478511569101286542578624291186 26; break;
case 86 : cos_a = -0.489292916933923602813755071690 58; break;
case 85 : cos_a = -0.5; break;
case 84 : cos_a = -0.510631193180906965938095877430; break;
case 83 : cos_a = -0.521184882876585074487701312465 66; break;
case 82 : cos_a = -0.531659467250436088005871177919 77; break;
case 81 : cos_a = -0.542053356472449348347950810836 73; break;

case 80 : cos_a = -0.552364972960505810763100522900 37; break;
case 79 : cos_a = -0.562592751619823095613603917564 83; break;
case 78 : cos_a = -0.572735140080505215003001280232 9; break;
case 77 : cos_a = -0.582790598933160919070096363827 79; break;
case 76 : cos_a = -0.592757601962554885034844283507; break;
case 75 : cos_a = -0.602634636379256389178588154986 84; break;
case 74 : cos_a = -0.612420203049249106206813687302; break;
case 73 : cos_a = -0.622112816721473898244443556800 3; break;
```

```
case 72 : cos_a = -0.631711006253250957262429149575l6; break;
case 71 : cos_a = -0.641213314833578366119851208635l8; break;

case 70 : cos_a = -0.650618300204242113720062533882l; break;
case 69 : cos_a = -0.652453487872260035788467694186; break;
case 68 : cos_a = -0.669130606358858213826273330686l8; break;
case 67 : cos_a = -0.678235117349233975989884596670l5; break;
case 66 : cos_a = -0.687236685969262716600193503847; break;
case 65 : cos_a = -0.696133945962926608280458080217l4; break;
case 64 : cos_a = -0.704925546906147157479095692186l7; break;
case 63 : cos_a = -0.713610154411752294620056051584l1; break;
case 62 : cos_a = -0.722186450332009335794134366402l3; break;
case 61 : cos_a = -0.730653132958693146264235824977l2; break;

case 60 : cos_a = -0.739008917220659115924534309872l5; break;
case 59 : cos_a = -0.747252534878890962498493415825l2; break;
case 58 : cos_a = -0.755382734718375816497527647948; break;
case 57 : cos_a = -0.763398282741102963065064056560l9; break;
case 56 : cos_a = -0.771297962347180641340113168482l4; break;
case 55 : cos_a = -0.779080574525670431924360620607l9; break;
case 54 : cos_a = -0.786744938033483247203053664633l1; break;
case 53 : cos_a = -0.794289889575286077788535454518l7; break;
case 52 : cos_a = -0.801714283980066691000680473351l4; break;
case 51 : cos_a = -0.809016437494742410229341718282; break;

case 50 : cos_a = -0.816196912356221690871852540431l1; break;
case 49 : cos_a = -0.823252948157587241631910290545l4; break;
case 48 : cos_a = -0.830184030815550642327580763126l8; break;
case 47 : cos_a = -0.836989108331977867601266425509l7; break;
case 46 : cos_a = -0.843667147833766335971938780883l2; break;
case 45 : cos_a = -0.850217135729614152134143922949l5; break;
case 44 : cos_a = -0.856638077863862761977140640587l5; break;
case 43 : cos_a = -0.862928966738967012132394687355; break;
case 42 : cos_a = -0.869088946305528317502054725493; break;
case 41 : cos_a = -0.875116982822266781090682667l9; break;

case 40 : cos_a = -0.881012194285784506008708817925l9; break;
case 39 : cos_a = -0.886773685920061910528494494573l5; break;
case 38 : cos_a = -0.892400583247947821468252054720l3; break;
case 37 : cos_a = -0.897892032220258091945357880940l3; break;
case 36 : cos_a = -0.903247134612887712448326961716; break;
case 35 : cos_a = -0.908465271819523686111503647585l2; break;
case 34 : cos_a = -0.913545457642600895502127571985l2; break;
case 33 : cos_a = -0.918486985745922989237182590606; break;
case 32 : cos_a = -0.923289106105489354018447523132l4; break;
case 31 : cos_a = -0.927951089856574643323333102751; break;

case 30 : cos_a = -0.932472229404355804573115891821l6; break;
case 29 : cos_a = -0.936851838531310605594497792201l8; break;
case 28 : cos_a = -0.941089252501371583201697812929l9; break;
case 27 : cos_a = -0.945183828160819530851973014721l1; break;
case 26 : cos_a = -0.949134944035901222439372824789l5; break;
case 25 : cos_a = -0.952942000427156555831028303415l6; break;
case 24 : cos_a = -0.956604419500440719379370338l2; break;
case 23 : cos_a = -0.960121645374628124775597137261l4; break;
case 22 : cos_a = -0.963493144205983118519285112411l6; break;
case 21 : cos_a = -0.966718404269187459620608601349l7; break;

case 20 : cos_a = -0.969796936035009471819536015653l6; break;
case 19 : cos_a = -0.972728272244604757758235765477l5; break;
case 18 : cos_a = -0.975511967980436639072509518262l2; break;
case 17 : cos_a = -0.978147600733805637928566747869l6; break;
case 16 : cos_a = -0.980634770468977750395941820011l7; break;
case 15 : cos_a = -0.982973068390177828194884485528; break;
case 14 : cos_a = -0.985162233467506503764287473294; break;
```

```
    case 13 : cos_a = -0.98720183955356901027473053727675; break;
    case 12 : cos_a = -0.98909160837114597349770837305954; break;
    case 11 : cos_a = -0.99083125309156026805871214960483; break;

    case 10 : cos_a = -0.99242050967193575826145605410729; break;
    case 9 : cos_a = -0.99385913689527366518807139262033; break;
    case 8 : cos_a = -0.99514691640706442737560041937711; break;
    case 7 : cos_a = -0.99628365274842949812969013953886; break;
    case 6 : cos_a = -0.99726917338578804922110119591952; break;
    case 5 : cos_a = -0.99810332873704407815955807227985; break;
    case 4 : cos_a = -0.99878599219428994437181274925287; break;
    case 3 : cos_a = -0.99931706014302288834641482622583; break;
    case 2 : cos_a = -0.99969645197787161706632560365192; break;
    case 1 : cos_a = -0.99992411011483056875185094912157; break;
    case 0 : cos_a = -1; break;
  }
}
/*============================================================*/
void tegangan_efektif() {
  V = (220 / 2) * (1 + cos_a); //
}
/*============================================================*/
void baca_suhu() {
  int dev = 0x5A << 1;
  int data_low = 0;
  int data_high = 0;
  int pec = 0;

  i2c_start_wait(dev + I2C_WRITE);
  i2c_write(0x07);

  // read
  i2c_rep_start(dev + I2C_READ);
  data_low = i2c_readAck();
  data_high = i2c_readAck();
  pec = i2c_readNak();
  i2c_stop();

  double tempFactor = 0.02;
  double tempData = 0x0000;
  int frac;

  tempData = (double)(((data_high & 0x007F) << 8) + data_low);
  tempData = (tempData * tempFactor) - 0.01;

  suhu = tempData - 273.15;
  float fahrenheit = (suhu * 1.8) + 32;


}
/*============================================================*/
void fuzy1() {
  KP = kp;
  KD = ki;
  KI = ki;
}
/*============================================================*/
void fuzy2() {
  KP = kp * 20;
  KD = ki * 20;
  KI = ki * 20;
}
/*============================================================*/
void pilih() {
  if (suhu >= suhu0) {
```

```
    a = 1;
    fuzy2();
    DETIK = DETIK + (ts * 0.001);
    detik = DETIK;
    jumlah_cuplik++;
    data_suhu = data_suhu + suhu;
  }
  else if (suhu < suhu0) {
    if (a == 0) {
      fuzy1();
      RISE_TIME = RISE_TIME + (ts * 0.001);
      rise_time = RISE_TIME;
      detik = 0; menit = 0; jam = 0;
    }
    else if (a == 1) {
      fuzy2();
      DETIK = DETIK + (ts * 0.001);
      detik = DETIK;
      jumlah_cuplik++;
      data_suhu = data_suhu + suhu;
    }
  }
}
/*============================================================*/
void kontrol_pid() {
  last_error = error;
  error = suhu0 - suhu;
  p = (KP * error);
  i = (KI / s) * (error + last_error);
  d = (KD * s) * (error - last_error);
  pid = (p + i + d);
  pwm = kecepatan + pid;

  if (pwm > 255)    {
    pwm = 255;
  }
  else if (pwm < 0) {
    pwm = 0;
  }
  analogWrite(heaterPin, pwm);
  angle();
  tegangan_efektif();
}
/*============================================================*/
void timer() {
  if (detik > 59) {
    DETIK  =  0;
    menit++;
    if (menit > 59) {
      menit  =  0;
      jam++;
      if (jam > 23) {
        jam  =  0;
      }
    }
  }
  TIME = jam + titik + menit + titik + detik + spasi;
}
/*============================================================*/
void LCD() {
  lcd.setCursor(0, 0);   lcd.print(strpwm);      lcd.print(pwm);
  lcd.print(spasi);
  lcd.setCursor(0, 1);   lcd.print(strtegangan); lcd.print(V);
  lcd.print(AC);
  lcd.setCursor(0, 2);   lcd.print(strsuhu);     lcd.print(suhu);
```

```
  lcd.write(1); lcd.print("C ");
  lcd.setCursor(0, 3);   lcd.print(strtime);     lcd.print(TIME);
}
/*=========================================================*/
void waktu_habis() {
  if (menit >= hour)
  {
    lcd.clear();
    analogWrite(heaterPin, 0);
    judul();
    lcd.setCursor(0, 1); lcd.print("    PROSES      ");
    lcd.setCursor(0, 2); lcd.print("    SELESAI!    ");
    selesai();
  }
}
/*=========================================================*/
void selesai()  {
  menu:
  tombol();
  inByte = BT.read();
  if (inByte == 'c') {
    hasil();
  }
  else if (OK == HIGH) {
    delay(250);
    lcd.clear();
    hasil();
  }
  else goto menu;
}
/*=========================================================*/
void hasil() {
  suhu_rerata = (data_suhu / jumlah_cuplik);

  if (suhu_rerata <= suhu0) {
    persentase = (suhu_rerata / suhu0) * 100;
  }
  else if (suhu_rerata > suhu0) {
    persentase = (2 - (suhu_rerata / suhu0)) * 100;
  }

  int SUHU_RERATA = suhu_rerata;
  int PERSENTASE = persentase;

  lcd.setCursor(0, 0);
  for (int i=0; i<16; i++) {
    lcd.write(5);
  }
  lcd.setCursor(0, 1);  lcd.print("RisTime "); lcd.print(rise_time);
  lcd.print("s  ");
  lcd.setCursor(0, 2);  lcd.print("TempAvg ");  lcd.print(suhu_rerata);
  lcd.write(1); lcd.print("C ");
  lcd.setCursor(0, 3);  lcd.print("Success ");  lcd.print(persentase);
  lcd.print("%    ");

  HASIL:
  inByte = BT.read();
  if (inByte == 'c') {
    setup();
  }
  tombol();
  if (OK == HIGH) {
    delay(250);
    setup();
  }
```

```
  else goto HASIL;
}
/*============================================================*/
void BT_rx() {
  inByte = BT.read();
  if (inByte == 'a') {
    mati();
  }

  tombol();
  if (CANCEL == HIGH) {
    mati();
  }
}
/*============================================================*/
void BT_tx() {
  BT.print(suhu); BT.println("'C");
}
/*============================================================*/
void mati() {
  lcd.clear();
  analogWrite(heaterPin, 0);
  judul();
  lcd.setCursor(0, 1); lcd.print("   PROSES       ");
  lcd.setCursor(0, 2); lcd.print("   DIHENTIKAN!  ");
  restart();
}
/*============================================================*/
void restart() {
  menu:
  tombol();
  inByte = BT.read();
  if (inByte == 'c') {
    setup();
  }
  else if (OK == HIGH) {
    delay(250);
    setup();
  }
  else goto menu;
}
/*============================================================*/
void tombol() {
  OK = digitalRead(okPin); CANCEL = digitalRead(cancelPin); UP =
digitalRead(upPin); DOWN = digitalRead(downPin);
}
/*============================================================*/
void tulis() {
  EEPROM.write(0, kp);
  EEPROM.write(1, ki);
  EEPROM.write(2, kd);
  EEPROM.write(3, suhu0);
  EEPROM.write(4, hour);
  EEPROM.write(5, kecepatan);
  EEPROM.write(6, (ts / 10));
  EEPROM.write(7, sampling);
}
/*============================================================*/
void baca() {
  kp = EEPROM.read(0);
  ki = EEPROM.read(1);
  kd = EEPROM.read(2);
  suhu0 = EEPROM.read(3);
  hour = EEPROM.read(4);
  kecepatan = EEPROM.read(5);
```

```
  ts = EEPROM.read(6) * 10;
  sampling = EEPROM.read(7);
}
/*============================================================*/
void loop() {
  cmilis = millis();
  while (cmilis - pmilis >= ts)
  {
    pmilis = cmilis;
  /*--------------------------*/
    baca_suhu();
    kontrol_pid();
    pilih();
    timer();
    LCD();
    BT_tx();
    BT_rx();
    waktu_habis();
  /*--------------------------*/
  }
}
```

2. Baris Program Arduino Nano

```
int AC_LOAD = 13;
int pwm = A0;
int T = 10000;
float Ton, Toff;

void setup()
{
  pinMode(AC_LOAD, OUTPUT);
  digitalWrite(2,HIGH);
  attachInterrupt(0, zero_crosss_int, RISING);
}

void zero_crosss_int() // {
  Ton  =  analogRead(pwm) * 9.76;
  if(Ton<300) {Ton=300;}
  if(Ton>=9800) {Ton=9800;}
  Toff = T-Ton;
  delayMicroseconds(Toff);
  digitalWrite(AC_LOAD, HIGH);
  delayMicroseconds(1);
  digitalWrite(AC_LOAD, LOW);
}

void loop()
{
}
```

3. Baris Program Telemetri untuk *Smartphone* Android